```
; ****************************************************************************
; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3
; ----------------------------------------------------------------------------
; Last Update: 02/03/2021
; ----------------------------------------------------------------------------
; 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[0A6E]                   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[066E]                   mov   [mem_1m_1k], cx
00000017 8916[086E]                   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[086E]                     mov    si, [mem_16m_64k]
00000076 8936[E90E]                     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[5C09]                       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[3743]                       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[EB0E]                     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[086E]                     mov    di, [mem_16m_64k]
000000DD 893E[E90E]                     mov    [real_mem_16m_64k], di

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

000000E5 8936[086E]                     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) - v2.0.2 - diskinit.s
                                      <1> ; ----------------------------------------------------------------------------
                                      <1> ; Last Update: 29/08/2020
                                      <1> ; ----------------------------------------------------------------------------
                                      <1> ; Beginning: 24/01/2016
                                      <1> ; ----------------------------------------------------------------------------
                                      <1> ; Assembler: NASM version 2.14 (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>       ; 29/08/2020
                                      <1>       ; 17/07/2020
                                      <1>       ; 14/07/2020 (TRDOS 386 v2.0.2)
                                      <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
```

```
                        <1>       inc    byte [hdc]   ; count of hard disks (EDD present)
                        <1>         mov     [last_drv], dl  ; last hard disk number
                        <1>       mov    bx, hd0_type - 80h
                        <1>       add    bx, dx
                        <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[0E6E]       <1>       mov    si, fd0_type
                        <1> L3:
                        <1>       ; 14/01/2015
00000118 8816[0B6E]     <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 E8DE01         <1>       call   set_disk_parms
                        <1>       ; 10/12/2014
00000127 81FE[0E6E]     <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:
00000132 B27F           <1>       mov    dl, 7Fh
                        <1>       ; 24/12/2014
00000134 803E[0D6E]00   <1>       cmp    byte [hdc], 0 ; EDD present or not ?
00000139 0F878700       <1>         ja      L10       ; yes, all fixed disk operations
                        <1>                           ; will be performed according to
                        <1>                           ; present EDD specification
                        <1>
                        <1>       ; 17/07/2020
                        <1>       ; Note: Virtual CPU will not come here while
                        <1>       ; running in QEMU, Bochs, VirtualBox emulators !!!
                        <1>
                        <1>       ; 17/07/2020
                        <1>       ; Older BIOS (INT 13h, AH = 48h is not available)
                        <1> L6:
0000013D FEC2           <1>       inc    dl
0000013F 8816[0B6E]     <1>         mov     [drv], dl
00000143 8816[0C6E]     <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 0F82A801       <1>         jc      L13       ; fixed disk drive not ready
0000014F 8816[0D6E]     <1>         mov     [hdc], dl ; number of drives
                        <1>       ;; 14/01/2013
                        <1>       ;;push cx
00000153 E8AF01         <1>       call   set_disk_parms
                        <1>       ;;pop  cx
                        <1>       ;
                        <1>       ;;and  cl, 3Fh        ; sectors per track (bits 0-6)
00000156 8A16[0B6E]     <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 0F857C01       <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>
                        <1>       ; 02/02/2015
                        <1>       ;mov   cl, [drv]
                        <1>       ;mov   bl, cl
                        <1>       ;mov   ax, 1F0h
                        <1>       ;and   bl, 1
                        <1>       ;jz    short L9
                        <1>       ;shl   bl, 4
                        <1>       ;sub   ax, 1F0h-170h
```

```
                                <1>
                                <1>       ; 17/07/2020
                                <1>       ; (Only 1F0h port address must be valid for old ROM BIOSes)
00000190 B8F001                 <1>       mov    ax, 1F0h
00000193 B3A0                   <1>       mov    bl, 0A0h
00000195 80FA80                 <1>       cmp    dl, 80h
00000198 7603                   <1>       jna    short L9
                                <1>       ; dl = 81h
0000019A 80C310                 <1>       add    bl, 10h  ; slave disk
                                <1>       ;sub   ax, 1F0h-170h
                                <1> L9:
0000019D AB                     <1>       stosw  ; I/O PORT Base Address (1F0h, 170h)
0000019E 050602                 <1>       add    ax, 206h
000001A1 AB                     <1>       stosw  ; CONTROL PORT Address (3F6h, 376h)
000001A2 88D8                   <1>       mov    al, bl  ; bit 4, master/slave disk bit
                                <1>       ;add   al, 0A0h ; 17/07/2020
000001A4 AA                     <1>       stosb  ; Device/Head Register upper nibble
                                <1>       ;
000001A5 FE06[0B6E]             <1>       inc    byte [drv]
000001A9 BB[906D]               <1>       mov    bx, hd0_type - 80h
000001AC 01CB                   <1>       add    bx, cx
000001AE 800F80                 <1>         or     byte [bx], 80h  ; present sign (when lower nibble is 0)
000001B1 A0[0D6E]               <1>       mov    al, [hdc]
000001B4 FEC8                   <1>       dec    al
000001B6 0F843D01               <1>         jz     L13
000001BA 80FA80                 <1>       cmp    dl, 80h
000001BD 0F867CFF               <1>         jna  L6 ; Max. 2 hard disks  ; 17/07/2020
000001C1 E93301                 <1>         jmp    L13
                                <1> L10:
000001C4 FEC2                   <1>       inc    dl
                                <1>       ; 25/12/2014
000001C6 8816[0B6E]             <1>       mov    [drv], dl
000001CA B408                   <1>       mov    ah, 08h ; Return drive parameters
000001CC CD13                   <1>       int    13h    ; (conventional function)
000001CE 0F822501               <1>         jc     L13
                                <1>       ; 14/01/2015
000001D2 8A16[0B6E]             <1>       mov    dl, [drv]
000001D6 52                     <1>       push   dx
000001D7 51                     <1>       push   cx
000001D8 E82A01                 <1>       call   set_disk_parms
000001DB 59                     <1>       pop    cx
000001DC 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
000001DD BE[5000]               <1>       mov    si, _int13h_48h_buffer
                                <1>       ; 09/07/2016
000001E0 B81E00                 <1>       mov    ax, 001Eh
000001E3 8824                   <1>       mov    [si], ah ; 0
000001E5 46                     <1>       inc    si
000001E6 8904                   <1>       mov    word [si], ax
                                <1>       ; word [si] = 30
                                <1>       ;
000001E8 B448                   <1>       mov    ah, 48h       ; Get drive parameters (EDD function)
000001EA CD13                   <1>       int    13h
000001EC 0F820701               <1>         jc     L13
                                <1>
                                <1>       ; 29/08/2020
                                <1>       ; 04/02/2016 (ebx -> bx)
                                <1>       ; 14/01/2015
000001F0 28FF                   <1>       sub    bh, bh
000001F2 88D3                   <1>       mov    bl, dl
                                <1>       ;sub   bl, 80h
                                <1>       ; 29/08/2020
000001F4 81C3[906D]             <1>       add    bx, (hd0_type - 80h)
                                <1>       ;mov   al, [bx]
000001F8 8A07                   <1>       mov    al, [bx]
000001FA 0C80                   <1>       or     al, 80h
000001FC 8807                   <1>       mov    [bx], al
000001FE 81EB[0E6E]             <1>       sub    bx, hd0_type - 2 ; 15/01/2015
                                <1>       ;add   bx, drv.status
                                <1>       ;mov   [bx], al
                                <1>       ; 29/08/2020
00000202 8887[5A6E]             <1>       mov    [bx+drv.status], al
                                <1>       ; 04/02/2016 (eax -> ax)
                                <1>       ;mov   ax, [si+16]
                                <1>       ; 14/07/2020
                                <1>       ;mov   di, [si+18]
                                <1>       ;;test ax, [si+18]
                                <1>       ;test  ax, di ; 14/07/2020
                                <1>       ;jz    short L10_A0h ; (!) ; 17/07/2020
                                <1>                     ; 'CHS only' disks on EDD system
                                <1>                     ;  are reported with ZERO disk size
                                <1>                     ; (if so, we must not overwrite
                                <1>                     ; calculated disk size in 'set_disk_parms')
                                <1>       ; 29/08/2020
00000206 8B4410                 <1>       mov    ax, [si+16]
00000209 8B7C12                 <1>       mov    di, [si+18]
0000020C 09C0                   <1>       or     ax, ax
0000020E 7504                   <1>       jnz    short L10_LBA
00000210 09FF                   <1>       or     di, di
00000212 740B                   <1>       jz     short L10_A0h
                                <1> L10_LBA:
                                <1>       ;sub   bx, drv.status
00000214 C1E302                 <1>       shl    bx, 2
                                <1>       ;add   bx, drv.size ; disk size (in sectors)
                                <1>       ;mov   [bx], ax
                                <1>       ; 29/08/2020
00000217 8987[3E6E]             <1>       mov    [bx+drv.size], ax
                                <1>       ;mov   ax, [si+18]
                                <1>       ;;mov  [bx], ax
                                <1>       ;mov   [bx+2], ax ; BugFix ; 15/07/2020
```

```
                                <1>       ; 14/07/2020
                                <1>       ;mov   [bx+2], di ; 15/07/2020
                                <1>       ; 29/08/2020
0000021B 89BF[406E]             <1>       mov    [bx+drv.size+2], di
                                <1> L10_A0h:
                                <1>       ; 17/07/2020
                                <1>       ; Note: Virtual CPU will jump here from above (!) test
                                <1>       ;      while running in QEMU
                                <1>
                                <1>       ; Jump here to fix a ZERO (LBA) disk size problem
                                <1>       ; for CHS disks (28/02/2015)
                                <1>
                                <1>       ; 30/12/2014
0000021F BF0000                 <1>       mov    di, HD0_DPT
00000222 88D0                   <1>       mov    al, dl
00000224 83E003                 <1>       and    ax, 3
00000227 C0E005                 <1>       shl    al, 5 ; * 32
0000022A 01C7                   <1>       add    di, ax
0000022C B80090                 <1>       mov    ax, DPT_SEGM
0000022F 8EC0                   <1>       mov    es, ax
                                <1>       ;
00000231 88E8                   <1>       mov    al, ch ; max. cylinder number (bits 0-7)
00000233 88CC                   <1>       mov    ah, cl
00000235 C0EC06                 <1>       shr    ah, 6  ; max. cylinder number (bits 8-9)
00000238 40                     <1>       inc    ax     ; logical cylinders (limit 1024)
00000239 AB                     <1>       stosw
0000023A 88F0                   <1>       mov    al, dh ; max. head number
                                <1>       ;
0000023C 30F6                   <1>       xor    dh, dh  ; 29/08/2020 (dh = 0 is needed here)
                                <1>       ;
0000023E FEC0                   <1>       inc    al
00000240 AA                     <1>       stosb         ; logical heads (limits 256)
00000241 B0A0                   <1>       mov    al, 0A0h ; Indicates translated table
00000243 AA                     <1>       stosb
00000244 8A440C                 <1>       mov    al, [si+12]
00000247 AA                     <1>       stosb         ; physical sectors per track
00000248 31C0                   <1>       xor    ax, ax
                                <1>       ;dec   ax      ; 02/01/2015
0000024A AB                     <1>       stosw         ; precompensation (obsolete)
                                <1>       ;xor   al, al  ; 02/01/2015
0000024B AA                     <1>       stosb         ; reserved
0000024C B008                   <1>       mov    al, 8   ; drive control byte
                                <1>                      ; (do not disable retries,
                                <1>                      ; more than 8 heads)
0000024E AA                     <1>       stosb
0000024F 8B4404                 <1>       mov    ax, [si+4]
00000252 AB                     <1>       stosw         ; physical number of cylinders
                                <1>       ;push  ax      ; 02/01/2015
00000253 8A4408                 <1>       mov    al, [si+8]
00000256 AA                     <1>       stosb         ; physical num. of heads (limit 16)
00000257 29C0                   <1>       sub    ax, ax
                                <1>       ;pop   ax      ; 02/01/2015
00000259 AB                     <1>       stosw         ; landing zone (obsolete)
0000025A 88C8                   <1>       mov    al, cl  ; logical sectors per track (limit 63)
0000025C 243F                   <1>       and    al, 3Fh
0000025E AA                     <1>       stosb
                                <1>       ;sub   al, al  ; checksum
                                <1>       ;stosb
                                <1>       ;
0000025F 83C61A                 <1>       add    si, 26   ; (BIOS) DPTE address pointer
00000262 AD                     <1>       lodsw
00000263 50                     <1>       push   ax ; *  ; (BIOS) DPTE offset
00000264 AD                     <1>       lodsw
00000265 50                     <1>       push   ax ; **       ; (BIOS) DPTE segment
                                <1>       ;
                                <1>       ; checksum calculation
00000266 89FE                   <1>       mov    si, di
00000268 06                     <1>       push   es
00000269 1F                     <1>       pop    ds
                                <1>       ;mov   cx, 16
0000026A B90F00                 <1>       mov    cx, 15
0000026D 29CE                   <1>       sub    si, cx
0000026F 30E4                   <1>       xor    ah, ah
                                <1>       ;del   cl
                                <1> L11:
00000271 AC                     <1>       lodsb
00000272 00C4                   <1>       add    ah, al
00000274 E2FB                   <1>       loop   L11
                                <1>       ;
00000276 88E0                   <1>       mov    al, ah
00000278 F6D8                   <1>       neg    al     ; -x+x = 0
0000027A AA                     <1>       stosb         ; put checksum in byte 15 of the tbl
                                <1>       ;
0000027B 1F                     <1>       pop    ds ; **       ; (BIOS) DPTE segment
0000027C 5E                     <1>       pop    si ; * ; (BIOS) DPTE offset
                                <1>       ;
                                <1>       ; 14/07/2020
                                <1>       ; 0FFFFh:0FFFFh = invalid DPTE address
0000027D 8B0C                   <1>       mov    cx, [si]
0000027F 8B4402                 <1>       mov    ax, [si+2]
00000282 21C1                   <1>       and    cx, ax
00000284 41                     <1>       inc    cx
00000285 7404                   <1>       jz     short L11c ; 0FFFFh:0FFFFh
00000287 0B04                   <1>       or     ax, [si]
00000289 752A                   <1>       jnz    short L11e ; <> 0
                                <1> L11c:
                                <1>       ; 17/07/2020
                                <1>       ; TRDOS 386 v2 DRVINIT assumptions:
                                <1>       ; (also by regarding QEMU, Bochs and VirtualBox settings)
                                <1>       ;      Hard disk 0 port address: 1F0h
                                <1>       ;      Hard disk 1 port address: 1F0h
                                <1>       ;      Hard disk 2 port address: 170h
                                <1>       ;      Hard disk 3 port address: 170h
                                <1>
                                <1>       ; in QEMU, hda=hd0 (1F0h) and hdb=hd1 (1F0h) -IRQ14-
```

```
                        <1>       ;      and hdc=hd2 (170h) and hdd=hd3 (170h) -IRQ15-
                        <1>
0000028B B8F001         <1>       mov    ax, 1F0h
                        <1>
                        <1>       ; 15/07/2020
                        <1>       ; 14/07/2020
                        <1>       ; Invalid DPTE address...
                        <1>       ; Default DPTE parms must be set for DISK_IO_CONT
                        <1>       ; (diskio.s)
                        <1>       ; 17/07/2020
                        <1>
                        <1>       ;mov   bl, dl
                        <1>       ;and   bl, 1
                        <1>       ;jz    short L11d
                        <1>
0000028E B3A0           <1>       mov    bl, 0A0h
                        <1>
00000290 F6C201         <1>       test   dl, 1
00000293 7403           <1>       jz     short L11g  ; Master (as default, for 80h & 82h))
                        <1>       ;shl   bl, 4 ; bl = 16 (bit 4 = 1 -> slave)
00000295 80C310         <1>       add    bl, 10h  ; Slave (as default, for 81h & 83h)
                        <1> L11g:
                        <1>       ; 17/07/2020
00000298 80FA82         <1>       cmp    dl, 82h      ; Hard disk 3 or 4 ?
0000029B 7203           <1>       jb     short L11d ; Primary ATA channel (hd0, hd1)
                        <1>                         ; (port address = 1F0h)
                        <1>
                        <1>       ; Secondary ATA channel (hd2, hd3)
                        <1>       ; (port address = 170h)
                        <1>
0000029D 2D8000         <1>       sub    ax, 1F0h-170h
                        <1> L11d:
                        <1>       ; 14/07/2020
000002A0 AB             <1>       stosw  ; I/O PORT Base Address (1F0h, 170h)
000002A1 050602         <1>       add    ax, 206h
000002A4 AB             <1>       stosw  ; CONTROL PORT Address (3F6h, 376h)
000002A5 88D8           <1>       mov    al, bl  ; Master/Slave bit (0 = Master)
                        <1>       ; 17/07/2020
                        <1>       ;or    al, 0A0h ; CHS (LBA enable bit = 0)
                        <1>                       ; (Bits 5&7, reserved bits  =  1)
000002A7 30E4           <1>       xor    ah, ah
                        <1>       ;stosb ; Device/Head Register upper nibble
000002A9 AB             <1>       stosw
000002AA 30C0           <1>       xor    al, al
000002AC B90500         <1>       mov    cx, 5
000002AF F3AB           <1>       rep    stosw ; clear remain part of the (fake) DPTE
000002B1 0E             <1>       push   cs
000002B2 1F             <1>       pop    ds
000002B3 EB2E           <1>       jmp    short L11f
                        <1> L11e:
                        <1>       ; 23/02/2015
000002B5 57             <1>       push   di
                        <1>       ; ES:DI points to DPTE (FDPTE) location
                        <1>       ;;mov  cx, 8
                        <1>       ;mov   cl, 8
000002B6 B90800         <1>       mov    cx, 8 ; 14/07/2020
000002B9 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)
000002BB 5B             <1>       pop    bx
000002BC 8CC8           <1>       mov    ax, cs
000002BE 8ED8           <1>       mov    ds, ax
000002C0 268B07         <1>       mov    ax, [es:bx]
000002C3 3DF001         <1>       cmp    ax, 1F0h
000002C6 7413           <1>       je     short L11a
000002C8 3D7001         <1>       cmp    ax, 170h
000002CB 740E           <1>       je     short L11a
                        <1>       ; invalidation
                        <1>       ; (because base port address is not 1F0h or 170h)
                        <1>       ;xor   bh, bh
                        <1>       ;mov   bl, dl
                        <1>       ; 29/08/2020
                        <1>       ;xor   dh, dh ; 0
000002CD 89D3           <1>       mov    bx, dx
                        <1>       ;sub   bl, 80h
                        <1>       ;mov   byte [bx+hd0_type], 0 ; not a valid disk drive !
                        <1>         ;or  byte [bx+drv.status+2], 0F0h ; (failure sign)
                        <1>       ; 29/08/2020
000002CF C687[906D]00   <1>       mov    byte [bx+hd0_type-80h], 0
000002D4 808F[DC6D]F0   <1>       or     byte [bx+drv.status-7Eh], 0F0h
000002D9 EB0F           <1>       jmp    short L11b
                        <1> L11a:
                        <1>       ; LBA validation
000002DB 268A4704       <1>       mov    al, [es:bx+4] ; Head register upper nibble
000002DF A840           <1>       test   al, 40h ; LBA bit (bit 6)
000002E1 7507           <1>       jnz    short L11b ; LBA type I/O is OK! (E0h or F0h)
                        <1> L11f:
                        <1>       ; force CHS type I/O for this drive (A0h or B0h)
                        <1>       ;sub   bh, bh
                        <1>       ;mov   bl, dl
                        <1>       ; 29/08/2020
                        <1>       ;xor   dh, dh ; 0
000002E3 89D3           <1>       mov    bx, dx
                        <1>       ;sub   bl, 80h ; 26/02/2015
                        <1>         ;and byte [bx+drv.status+2], 0FEh ; clear bit 0
                        <1>                         ; bit 0 = LBA ready bit
                        <1>       ; 29/08/2020
000002E5 80A7[DC6D]FE   <1>       and    byte [bx+drv.status-7Eh], 0FEh
                        <1>       ; 'diskio' procedure will check this bit !
                        <1> L11b:
000002EA 3A16[0C6E]     <1>       cmp    dl, [last_drv] ; 25/12/2014
000002EE 7307           <1>         jnb     short L13
```

```
000002F0 E9D1FE             <1>         jmp     L10
                            <1>
                            <1> L12:
                            <1>         ; Restore data registers
000002F3 8CC8               <1>         mov     ax, cs
000002F5 8ED8               <1>         mov     ds, ax
                            <1> L13:
                            <1>         ; 13/12/2014
000002F7 0E                 <1>         push    cs
000002F8 07                 <1>         pop     es
                            <1> L14:
                            <1>         ; clear keyboard buffer
000002F9 B411               <1>         mov     ah, 11h
000002FB CD16               <1>         int     16h
000002FD 744D               <1>         jz      short L16 ; no keys in keyboard buffer
000002FF B010               <1>         mov     al, 10h
00000301 CD16               <1>         int     16h
00000303 EBF4               <1>         jmp     short L14
                            <1>
                            <1> set_disk_parms:
                            <1>         ; 29/08/2020
                            <1>         ; 04/02/2016 (ebx -> bx)
                            <1>         ; 10/07/2015
                            <1>         ; 14/01/2015
                            <1>         ;push  bx
00000305 28FF               <1>         sub     bh, bh
00000307 8A1E[0B6E]         <1>         mov     bl, [drv]
0000030B 80FB80             <1>         cmp     bl, 80h
0000030E 7203               <1>         jb      short sdp0
00000310 80EB7E             <1>         sub     bl, 7Eh
                            <1> sdp0:
                            <1>         ;add   bx, drv.status
                            <1>         ;mov   byte [bx], 80h ; 'Present' flag
                            <1>         ; 29/08/2020
00000313 C687[5A6E]80       <1>         mov     byte [bx+drv.status], 80h
                            <1>         ;
00000318 88E8               <1>         mov     al, ch ; last cylinder (bits 0-7)
0000031A 88CC               <1>         mov     ah, cl ;
0000031C C0EC06             <1>         shr     ah, 6  ; last cylinder (bits 8-9)
                            <1>         ;sub   bx, drv.status
0000031F D0E3               <1>         shl     bl, 1
                            <1>         ;add   bx, drv.cylinders
00000321 40                 <1>         inc     ax  ; convert max. cyl number to cyl count
                            <1>         ;mov   [bx], ax
                            <1>         ; 29/08/2020
00000322 8987[146E]         <1>         mov     [bx+drv.cylinders], ax
00000326 50                 <1>         push    ax ; ** cylinders
                            <1>         ;sub   bx, drv.cylinders
                            <1>         ;add   bx, drv.heads
00000327 30E4               <1>         xor     ah, ah
00000329 88F0               <1>         mov     al, dh ; heads
0000032B 40                 <1>         inc     ax
                            <1>         ;mov   [bx], ax
                            <1>         ; 29/08/2020
0000032C 8987[226E]         <1>           mov  [bx+drv.heads], ax
                            <1>         ;sub     bx, drv.heads
                            <1>           ;add     bx, drv.spt
00000330 30ED               <1>         xor     ch, ch
00000332 80E13F             <1>         and     cl, 3Fh       ; sectors (bits 0-6)
                            <1>         ;mov   [bx], cx
                            <1>           ; 29/08/2020
00000335 898F[306E]         <1>         mov     [bx+drv.spt], cx
                            <1>         ;sub   bx, drv.spt
00000339 D1E3               <1>         shl     bx, 1
                            <1>         ;add   bx, drv.size ; disk size (in sectors)
                            <1>         ; LBA size = cylinders * heads * secpertrack
0000033B F7E1               <1>         mul     cx
0000033D 89C2               <1>         mov     dx, ax ; heads*spt
0000033F 58                 <1>         pop     ax ; ** cylinders
00000340 48                 <1>         dec     ax ; 1 cylinder reserved (!?)
00000341 F7E2               <1>         mul     dx ; cylinders * (heads*spt)
                            <1>         ;mov   [bx], ax
                            <1>         ;mov   [bx+2], dx
                            <1>         ; 29/08/2020
00000343 8987[3E6E]         <1>         mov     [bx+drv.size], ax
00000347 8997[406E]         <1>         mov     [bx+drv.size+2], dx
                            <1>         ;
                            <1>         ;pop   bx
0000034B C3                 <1>         retn
                            <1>
                            <1> L16:  ; 28/05/2016
                                      ;
                                      ; 10/11/2014
0000034C FA                           cli     ; Disable interrupts (clear interrupt flag)
                                              ; Reset Interrupt MASK Registers (Master&Slave)
                                      ;mov   al, 0FFh      ; mask off all interrupts
                                      ;out   21h, al              ; on master PIC (8259)
                                      ;jmp   $+2  ; (delay)
                                      ;out   0A1h, al      ; on slave PIC (8259)
                                      ;
                                      ; Disable NMI
0000034D B080                         mov     al, 80h
0000034F E670                         out     70h, al             ; set bit 7 to 1 for disabling NMI
                                      ;23/02/2015
                                      ;nop                 ;
                                      ;in    al, 71h              ; read in 71h just after writing out to 70h
                                                           ; for preventing unknown state (!?)
                                      ;
                                      ; 20/08/2014
                                      ; Moving the kernel 64 KB back (to physical address 0)
                                      ; DS = CS = 1000h
                                      ; 05/11/2014
00000351 31C0                         xor     ax, ax
00000353 8EC0                         mov     es, ax ; ES = 0
                                      ;
```

```
                                        ; 04/07/2016 - TRDOS 386 (64K - 128K kernel)
00000355 31F6                                 xor    si, si
00000357 31FF                           xor    di, di
00000359 B90040                         mov    cx, 16384
0000035C F366A5                         rep    movsd
                                        ;
0000035F 06                             push   es ; 0
00000360 68[6403]                       push   L17
00000363 CB                             retf
                                    L17:
00000364 B90010                         mov    cx, 1000h
00000367 8EC1                           mov    es, cx  ; 1000h
00000369 01C9                           add    cx, cx
0000036B 8ED9                           mov    ds, cx  ; 2000h
0000036D 29F6                           sub    si, si
0000036F 29FF                           sub    di, di
00000371 B90040                         mov    cx, 16384
00000374 F366A5                         rep    movsd

                                        ; Turn off the floppy drive motor
00000377 BAF203                           mov     dx, 3F2h
0000037A EE                               out     dx, al ; 0 ; 31/12/2013

                                        ; Enable access to memory above one megabyte
                                    L18:
0000037B E464                           in     al, 64h
0000037D A802                           test   al, 2
0000037F 75FA                             jnz     short L18
00000381 B0D1                           mov    al, 0D1h     ; Write output port
00000383 E664                           out    64h, al
                                    L19:
00000385 E464                           in     al, 64h
00000387 A802                           test   al, 2
00000389 75FA                             jnz     short L19
0000038B B0DF                           mov    al, 0DFh     ; Enable A20 line
0000038D E660                           out    60h, al
                                   ;L20:
                                        ;
                                        ; Load global descriptor table register

                                          ;mov     ax, cs
                                          ;mov     ds, ax

0000038F 2E0F0116[786D]                   lgdt    [cs:gdtd]

00000395 0F20C0                           mov     eax, cr0
                                        ; or   eax, 1
00000398 40                             inc     ax
00000399 0F22C0                         mov     cr0, eax

                                        ; Jump to 32 bit code

0000039C 66                             db 66h                  ; Prefix for 32-bit
0000039D EA                             db 0EAh             ; Opcode for far jump
0000039E [A4030000]                     dd StartPM          ; Offset to start, 32-bit
                                                            ; (1000h:StartPM = StartPM + 10000h)
000003A2 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
000003A4 66B81000                       mov ax, KDATA           ; Save data segment identifier
000003A8 8ED8                             mov ds, ax              ; Move a valid data segment into DS register
000003AA 8EC0                                mov es, ax              ; Move data segment into ES register
000003AC 8EE0                                mov fs, ax              ; Move data segment into FS register
000003AE 8EE8                                mov gs, ax              ; Move data segment into GS register
000003B0 8ED0                             mov ss, ax              ; Move data segment into SS register
000003B2 BC00000900                         mov esp, 90000h         ; Move the stack pointer to 090000h

                                   clear_bss: ; Clear uninitialized data area
                                        ; 11/03/2015
000003B7 31C0                           xor  eax, eax ; 0
000003B9 B9E0620000                     mov  ecx, (bss_end - bss_start)/4
                                        ;shr  ecx, 2 ; bss section is already aligned for double words
000003BE BF[2A870100]                   mov  edi, bss_start
000003C3 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
000003C5 B108                           mov    cl, 8
000003C7 BF00001000                     mov    edi, MEM_ALLOC_TBL
000003CC F3AB                           rep    stosd           ; clear Memory Allocation Table
                                                               ; for the first 1 MB memory
                                        ;
000003CE 668B0D[066E0000]               mov    cx, [mem_1m_1k]        ; Number of contiguous KB between
                                                               ; 1 and 16 MB, max. 3C00h = 15 MB.
000003D5 66C1E902                       shr    cx, 2           ; convert 1 KB count to 4 KB count
000003D9 890D[208A0100]                 mov    [free_pages], ecx
000003DF 668B15[086E0000]               mov    dx, [mem_16m_64k]  ; Number of contiguous 64 KB blocks
                                                               ; between 16 MB and 4 GB.
```

```
000003E6 6609D2                 or      dx, dx
000003E9 7413                   jz      short mi_0
                                ;
000003EB 6689D0                 mov     ax, dx
000003EE C1E004                 shl     eax, 4          ; 64 KB -> 4 KB (page count)
000003F1 0105[208A0100]         add     [free_pages], eax
000003F7 0500100000             add     eax, 4096       ; 16 MB = 4096 pages
000003FC EB07                   jmp     short mi_1
                        mi_0:
000003FE 6689C8                 mov     ax, cx
00000401 66050001               add     ax, 256             ; add 256 pages for the first 1 MB
                        mi_1:
00000405 A3[1C8A0100]           mov     [memory_size], eax ; Total available memory in pages
                                                           ; 1 alloc. tbl. bit = 1 memory page
                                                           ; 32 allocation bits = 32 mem. pages
                                ;
0000040A 05FF7F0000             add     eax, 32767      ; 32768 memory pages per 1 M.A.T. page
0000040F 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
00000412 66A3[308A0100]         mov     [mat_size], ax          ; Memory Alloc. Table Size in pages
00000418 C1E00C                 shl     eax, 12                 ; 1 M.A.T. page = 4096 bytes
                                ;                       ; Max. 32 M.A.T. pages (4 GB memory)
0000041B 89C3                   mov     ebx, eax        ; M.A.T. size in bytes
                                ; Set/Calculate Kernel's Page Directory Address
0000041D 81C300001000           add     ebx, MEM_ALLOC_TBL
00000423 891D[188A0100]         mov     [k_page_dir], ebx  ; Kernel's Page Directory address
                                                           ; just after the last M.A.T. page
                                ;
00000429 83E804                 sub     eax, 4          ; convert M.A.T. size to offset value
0000042C A3[288A0100]           mov     [last_page], eax   ; last page ofset in the M.A.T.
                                ;                       ; (allocation status search must be
                                                           ; stopped after here)
00000431 31C0                   xor     eax, eax
00000433 48                     dec     eax             ; FFFFFFFFh (set all bits to 1)
00000434 6651                   push    cx
00000436 C1E905                 shr     ecx, 5          ; convert 1 - 16 MB page count to
                                                           ; count of 32 allocation bits
00000439 F3AB                   rep     stosd
0000043B 6659                   pop     cx
0000043D 40                     inc     eax             ; 0
0000043E 80E11F                 and     cl, 31          ; remain bits
00000441 7412                   jz      short mi_4
00000443 8907                   mov     [edi], eax      ; reset
                        mi_2:
00000445 0FAB07                 bts     [edi], eax      ; 06/11/2014
00000448 FEC9                   dec     cl
0000044A 7404                   jz      short mi_3
0000044C FEC0                   inc     al
0000044E EBF5                   jmp     short mi_2
                        mi_3:
00000450 28C0                   sub     al, al          ; 0
00000452 83C704                 add     edi, 4          ; 15/11/2014
                        mi_4:
00000455 6609D2                 or      dx, dx         ; check 16M to 4G memory space
00000458 7421                   jz      short mi_6     ; max. 16 MB memory, no more...
                                ;
0000045A B900021000             mov     ecx, MEM_ALLOC_TBL + 512 ; End of first 16 MB memory
                                ;
0000045F 29F9                   sub     ecx, edi       ; displacement (to end of 16 MB)
00000461 7406                   jz      short mi_5     ; jump if EDI points to
                                                       ;         end of first 16 MB
00000463 D1E9                   shr     ecx, 1         ; convert to dword count
00000465 D1E9                   shr     ecx, 1         ; (shift 2 bits right)
00000467 F3AB                   rep     stosd          ; reset all bits for reserved pages
                                                       ; (memory hole under 16 MB)
                        mi_5:
00000469 6689D1                 mov     cx, dx         ; count of 64 KB memory blocks
0000046C D1E9                   shr     ecx, 1         ; 1 alloc. dword per 128 KB memory
0000046E 9C                     pushf                  ; 16/11/2014
0000046F 48                     dec     eax            ; FFFFFFFFh (set all bits to 1)
00000470 F3AB                   rep     stosd
00000472 40                     inc     eax            ; 0
00000473 9D                     popf                   ; 16/11/2014
00000474 7305                   jnc     short mi_6
00000476 6648                   dec     ax             ; eax = 0000FFFFh
00000478 AB                     stosd
00000479 6640                   inc     ax             ; 0
                        mi_6:
0000047B 39DF                   cmp     edi, ebx       ; check if EDI points to
0000047D 730A                   jnb     short mi_7     ; end of memory allocation table
                                ;                      ; (>= MEM_ALLOC_TBL + 4906)
0000047F 89D9                   mov     ecx, ebx       ; end of memory allocation table
00000481 29F9                   sub     ecx, edi       ; convert displacement/offset
00000483 D1E9                   shr     ecx, 1         ; to dword count
00000485 D1E9                   shr     ecx, 1         ; (shift 2 bits right)
00000487 F3AB                   rep     stosd          ; reset all remain M.A.T. bits
                        mi_7:
                                ; Reset M.A.T. bits in M.A.T. (allocate M.A.T. pages)
00000489 BA00001000             mov     edx, MEM_ALLOC_TBL
                                ;sub    ebx, edx       ; Mem. Alloc. Tbl. size in bytes
                                ;shr    ebx, 12                ; Mem. Alloc. Tbl. size in pages
0000048E 668B0D[308A0100]       mov     cx, [mat_size]         ; Mem. Alloc. Tbl. size in pages
00000495 89D7                   mov     edi, edx
00000497 C1EF0F                 shr     edi, 15                ; convert M.A.T. address to
                                                           ; byte offset in M.A.T.
                                                           ; (1 M.A.T. byte points to
                                                           ;         32768 bytes)
                                                           ; Note: MEM_ALLOC_TBL address
                                                           ; must be aligned on 128 KB
                                                           ; boundary!
0000049A 01D7                   add     edi, edx       ; points to M.A.T.'s itself
                                ; eax = 0
0000049C 290D[208A0100]         sub     [free_pages], ecx ; 07/11/2014
                        mi_8:
```

```
607 000004A2 0FB307                     btr    [edi], eax     ; clear bit 0 to bit x (1 to 31)
608                                     ;dec   bl
609 000004A5 FEC9                       dec    cl
610 000004A7 7404                       jz     short mi_9
611 000004A9 FEC0                       inc    al
612 000004AB EBF5                       jmp    short mi_8
613                                 mi_9:
614                                     ;
615                                     ; Reset Kernel's Page Dir. and Page Table bits in M.A.T.
616                                     ;            (allocate pages for system page tables)
617
618                                     ; edx = MEM_ALLOC_TBL
619 000004AD 8B0D[1C8A0100]             mov    ecx, [memory_size] ; memory size in pages (PTEs)
620 000004B3 81C1FF030000               add    ecx, 1023      ; round up (1024 PTEs per table)
621 000004B9 C1E90A                     shr    ecx, 10              ; convert memory page count to
622                                                            ; page table count (PDE count)
623                                     ;
624 000004BC 51                         push   ecx            ; (**) PDE count (<= 1024)
625                                     ;
626 000004BD 41                         inc    ecx            ; +1 for kernel page directory
627                                     ;
628 000004BE 290D[208A0100]             sub    [free_pages], ecx ; 07/11/2014
629                                     ;
630 000004C4 8B35[188A0100]             mov    esi, [k_page_dir] ; Kernel's Page Directory address
631 000004CA C1EE0C                     shr    esi, 12              ; convert to page number
632                                 mi_10:
633 000004CD 89F0                       mov    eax, esi       ; allocation bit offset
634 000004CF 89C3                       mov    ebx, eax
635 000004D1 C1EB03                     shr    ebx, 3         ; convert to alloc. byte offset
636 000004D4 80E3FC                     and    bl,  0FCh      ; clear bit 0 and bit 1
637                                                           ;   to align on dword boundary
638 000004D7 83E01F                     and    eax, 31              ; set allocation bit position
639                                                           ;  (bit 0 to bit 31)
640                                     ;
641 000004DA 01D3                       add    ebx, edx       ; offset in M.A.T. + M.A.T. address
642                                     ;
643 000004DC 0FB303                     btr    [ebx], eax     ; reset relevant bit (0 to 31)
644                                     ;
645 000004DF 46                         inc    esi            ; next page table
646 000004E0 E2EB                       loop   mi_10          ; allocate next kernel page table
647                                                           ; (ecx = page table count + 1)
648                                     ;
649 000004E2 59                         pop    ecx            ; (**) PDE count (= pg. tbl. count)
650                                     ;
651                                     ; Initialize Kernel Page Directory and Kernel Page Tables
652                                     ;
653                                     ; Initialize Kernel's Page Directory
654 000004E3 8B3D[188A0100]             mov    edi, [k_page_dir]
655 000004E9 89F8                       mov    eax, edi
656 000004EB 0C03                       or     al, PDE_A_PRESENT + PDE_A_WRITE
657                                                           ; supervisor + read&write + present
658 000004ED 89CA                       mov    edx, ecx       ; (**) PDE count (= pg. tbl. count)
659                                 mi_11:
660 000004EF 0500100000                 add    eax, 4096      ; Add page size (PGSZ)
661                                                            ; EAX points to next page table
662 000004F4 AB                         stosd
663 000004F5 E2F8                       loop   mi_11
664 000004F7 29C0                       sub    eax, eax       ; Empty PDE
665 000004F9 66B90004                   mov    cx, 1024       ; Entry count (PGSZ/4)
666 000004FD 29D1                       sub    ecx, edx
667 000004FF 7402                       jz     short mi_12
668 00000501 F3AB                       rep    stosd          ; clear remain (empty) PDEs
669                                     ;
670                                     ; Initialization of Kernel's Page Directory is OK, here.
671                                 mi_12:
672                                     ; Initialize Kernel's Page Tables
673                                     ;
674                                     ; (EDI points to address of page table 0)
675                                     ; eax = 0
676 00000503 8B0D[1C8A0100]             mov    ecx, [memory_size] ; memory size in pages
677 00000509 89CA                       mov    edx, ecx       ; (***)
678 0000050B B003                       mov    al, PTE_A_PRESENT + PTE_A_WRITE
679                                                           ; supervisor + read&write + present
680                                 mi_13:
681 0000050D AB                         stosd
682 0000050E 0500100000                 add    eax, 4096
683 00000513 E2F8                       loop   mi_13
684 00000515 6681E2FF03                 and    dx, 1023       ; (***)
685 0000051A 740B                       jz     short mi_14
686 0000051C 66B90004                   mov    cx, 1024
687 00000520 6629D1                     sub    cx, dx         ; from dx (<= 1023) to 1024
688 00000523 31C0                       xor    eax, eax
689 00000525 F3AB                       rep    stosd          ; clear remain (empty) PTEs
690                                                           ; of the last page table
691                                 mi_14:
692                                     ;  Initialization of Kernel's Page Tables is OK, here.
693                                     ;
694 00000527 89F8                       mov    eax, edi       ; end of the last page table page
695                                                            ; (beginging of user space pages)
696 00000529 C1E80F                     shr    eax, 15              ; convert to M.A.T. byte offset
697 0000052C 24FC                       and    al, 0FCh       ; clear bit 0 and bit 1 for
698                                                           ; aligning on dword boundary
699
700 0000052E A3[2C8A0100]               mov    [first_page], eax
701 00000533 A3[248A0100]               mov    [next_page], eax ; The first free page pointer
702                                                            ; for user programs
703                                                            ; (Offset in Mem. Alloc. Tbl.)
704                                     ;
705                                     ; Linear/FLAT (1 to 1) memory paging for the kernel is OK, here.
706                                     ;
707
708                                     ; Enable paging
709                                     ;
710 00000538 A1[188A0100]                 mov     eax, [k_page_dir]
711 0000053D 0F22D8                     mov    cr3, eax
```

```
712 00000540 0F20C0                     mov    eax, cr0
713 00000543 0D00000080                 or     eax, 80000000h      ; set paging bit (bit 31)
714 00000548 0F22C0                     mov    cr0, eax
715                                       ;jmp    KCODE:StartPMP
716
717 0000054B EA                         db 0EAh             ; Opcode for far jump
718 0000054C [52050000]                   dd StartPMP             ; 32 bit offset
719 00000550 0800                       dw KCODE            ; kernel code segment descriptor
720
721                                   StartPMP:
722                                         ; 06/11//2014
723                                         ; Clear video page 0
724                                         ;
725                                         ; Temporary Code
726                                         ;
727 00000552 B9E8030000                 mov    ecx, 80*25/2
728 00000557 BF00800B00                 mov    edi, 0B8000h
729                                         ; 30/01/2016
730                                         ;xor   eax, eax     ; black background, black fore color
731 0000055C B800070007                 mov    eax, 07000700h  ; black background, light gray fore color
732 00000561 F3AB                       rep    stosd
733
734                                         ; 19/08/2014
735                                         ; Kernel Base Address = 0
736                                         ; It is mapped to (physically) 0 in the page table.
737                                         ; So, here is exactly 'StartPMP' address.
738
739                                         ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
740 00000563 BE[864D0100]               mov    esi, starting_msg
741                                         ;; 14/08/2015 (kernel version message will appear
742                                         ;;            when protected mode and paging is enabled)
743 00000568 BF00800B00                 mov    edi, 0B8000h ; 27/08/2014
744
745                                         ; 30/11/2020
746                                         ; 14/11/2020 (TRDOS 386 v2.0.3)
747                                         ;cmp  byte [vbe3], 3 ; 03h
748                                         ;jne  short pkv_1
749                                         ;;mov  ah, 0Bh ; Black background, light cyan forecolor
750                                         ;; Light red TRDOS 386 version text shows VBE3 is ready !
751                                         ;mov  ah, 0Ch ; Black background, light red forecolor
752                                         ;jmp  short pkv_2
753                                   ;pkv_1:
754 0000056D B40A                         mov    ah, 0Ah ; Black background, light green forecolor
755                                   ;pkv_2:
756                                         ; 20/08/2014
757 0000056F E8DE030000                 call   printk
758
759                                         ; 'UNIX v7/x86' source code by Robert Nordier (1999)
760                                         ; // Set IRQ offsets
761                                         ;
762                                         ;  Linux (v0.12) source code by Linus Torvalds (1991)
763                                         ;
764                                                                 ;; ICW1
765 00000574 B011                         mov    al, 11h                 ; Initialization sequence
766 00000576 E620                       out    20h, al                 ;      8259A-1
767                                         ; jmp  $+2
768 00000578 E6A0                       out    0A0h, al            ;      8259A-2
769                                                                 ;; ICW2
770 0000057A B020                       mov    al, 20h                 ; Start of hardware ints (20h)
771 0000057C E621                       out    21h, al                 ;      for 8259A-1
772                                         ; jmp  $+2
773 0000057E B028                       mov    al, 28h                 ; Start of hardware ints (28h)
774 00000580 E6A1                       out    0A1h, al            ;      for 8259A-2
775                                                                 ;
776 00000582 B004                       mov    al, 04h                 ;; ICW3
777 00000584 E621                       out    21h, al                 ;      IRQ2 of 8259A-1 (master)
778                                         ; jmp  $+2
779 00000586 B002                       mov    al, 02h                 ;      is 8259A-2 (slave)
780 00000588 E6A1                       out    0A1h, al            ;
781                                                                 ;; ICW4
782 0000058A B001                       mov    al, 01h                 ;
783 0000058C E621                       out    21h, al                 ;      8086 mode, normal EOI
784                                         ; jmp  $+2
785 0000058E E6A1                       out    0A1h, al            ;      for both chips.
786
787                                         ;mov   al, 0FFh     ; mask off all interrupts for now
788                                         ;out   21h, al
789                                         ;; jmp       $+2
790                                         ;out   0A1h, al
791
792                                         ; 02/04/2015
793                                         ; 26/03/2015 System call (INT 30h) modification
794                                         ;  DPL = 3 (Interrupt service routine can be called from user mode)
795                                         ;
796                                         ;; Linux (v0.12) source code by Linus Torvalds (1991)
797                                         ;  setup_idt:
798                                         ;
799                                           ;; 16/02/2015
800                                         ;;mov     dword [DISKETTE_INT], fdc_int ; IRQ 6 handler
801                                         ; 21/08/2014 (timer_int)
802 00000590 BE[244A0100]                 mov    esi, ilist
803 00000595 8D3D[30870100]              lea    edi, [idt]
804                                         ; 26/03/2015
805 0000059B B930000000                 mov    ecx, 48             ; 48 hardware interrupts (INT 0 to INT 2Fh)
806                                         ; 02/04/2015
807 000005A0 BB00000800                 mov    ebx,  80000h
808                                   rp_sidt1:
809 000005A5 AD                         lodsd
810 000005A6 89C2                       mov    edx, eax
811 000005A8 66BA008E                   mov    dx, 8E00h
812 000005AC 6689C3                     mov    bx, ax
813 000005AF 89D8                       mov    eax, ebx     ; /* selector = 0x0008 = cs */
814                                                                 ; /* interrupt gate - dpl=0, present */
815 000005B1 AB                         stosd  ; selector & offset bits 0-15
816 000005B2 89D0                       mov    eax, edx
```

```
000005B4 AB                             stosd  ; attributes & offset bits 16-23
000005B5 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)
000005B7 B120                           mov    cl, 32       ; 32 software interrupts (INT 30h to INT 4Fh)
                                rp_sidt2:
000005B9 AD                             lodsd
000005BA 21C0                           and    eax, eax
000005BC 7413                           jz     short rp_sidt3
000005BE 89C2                           mov    edx, eax
000005C0 66BA00EE                       mov    dx, 0EE00h    ; P=1b/DPL=11b/01110b
000005C4 6689C3                         mov    bx, ax
000005C7 89D8                           mov    eax, ebx      ; selector & offset bits 0-15
000005C9 AB                             stosd
000005CA 89D0                           mov    eax, edx
000005CC AB                             stosd
000005CD E2EA                           loop   rp_sidt2
000005CF EB16                           jmp    short sidt_OK
                                rp_sidt3:
000005D1 B8[7D0D0000]                   mov    eax, ignore_int
000005D6 89C2                           mov    edx, eax
000005D8 66BA00EE                       mov    dx, 0EE00h    ; P=1b/DPL=11b/01110b
000005DC 6689C3                         mov    bx, ax
000005DF 89D8                           mov    eax, ebx      ; selector & offset bits 0-15
                                rp_sidt4:
000005E1 AB                             stosd
000005E2 92                             xchg   eax, edx
000005E3 AB                             stosd
000005E4 92                             xchg   edx, eax
000005E5 E2FA                           loop   rp_sidt4
                                sidt_OK:
000005E7 0F011D[7E6D0000]               lidt   [idtd]
                                        ;
                                        ; TSS descriptor setup ; 24/03/2015
000005EE B8[B0890100]                   mov    eax, task_state_segment
000005F3 66A3[2A6D0000]                 mov    [gdt_tss0], ax
000005F9 C1C010                         rol    eax, 16
000005FC A2[2C6D0000]                   mov    [gdt_tss1], al
00000601 8825[2F6D0000]                 mov    [gdt_tss2], ah
00000607 66C705[168A0100]68-            mov    word [tss.IOPB], tss_end - task_state_segment
0000060F 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)
00000610 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.)
00000614 0F00D8                         ltr    ax  ; Load task register
                                        ;
                                esp0_set0:
                                        ; 30/07/2015
00000617 8B0D[1C8A0100]                 mov    ecx, [memory_size] ; memory size in pages
0000061D C1E10C                         shl    ecx, 12 ; convert page count to byte count
00000620 81F900004000                   cmp    ecx, CORE ; beginning of user's memory space (400000h)
                                                         ; (kernel mode virtual address)
00000626 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)
00000628 B900004000                     mov    ecx, CORE
                                esp0_set1:
0000062D 89CC                           mov    esp, ecx ; top of kernel stack (**tss.esp0**)
                                esp0_set_ok:
                                        ; 30/07/2015 (**tss.esp0**)
0000062F 8925[B4890100]                 mov    [tss.esp0], esp
00000635 66C705[B8890100]10-              mov     word [tss.ss0], KDATA
0000063D 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!)
                                        ;
0000063E 30C0                           xor    al, al       ; Enable all hardware interrupts!
00000640 E621                           out    21h, al             ; (IBM PC-AT compatibility)
00000642 EB00                           jmp    $+2          ; (All conventional PC-AT hardware
00000644 E6A1                           out    0A1h, al     ;  interrupts will be in use.)
                                                            ; (Even if related hardware component
                                                            ;  does not exist!)
                                        ; Enable NMI
00000646 B07F                           mov    al, 7Fh             ; Clear bit 7 to enable NMI (again)
00000648 E670                           out    70h, al
                                        ; 23/02/2015
0000064A 90                             nop
0000064B 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
0000064D 6631DB                         xor    bx, bx
00000650 66BA0002                       mov    dx, 0200h    ; Row 2, column 0  ; 07/03/2015
00000654 E8461D0000                     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
00000659 803D[5C090000]02           cmp    byte [vbe3], 2 ; 02h
00000660 770B                       ja     short vbe3_pmid_chk
                                    ;jb    short display_mem_info
00000662 0F82CA010000               jb     display_mem_info ; 02/12/2020
00000668 E9FE010000                 jmp    check_boch_plex86_vbe
                                vbe3_pmid_chk:
0000066D B9EA7F0000                 mov    ecx, 32768 - (20+2) ; 32766 - PMInfoBlockSize
00000672 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
00000677 813E504D4944               cmp    dword [esi], 'PMID'
                                    ;je    short display_vbios_product_name
0000067D 740E                       je     short verify_pmib_chksum ; 15/11/2020
                                ;chk_pmi_sign_next:
0000067F 46                         inc    esi  ; inc si
00000680 E2F5                       loop   chk_pmi_sign

                                not_valid_pmib:
00000682 FE0D[5C090000]             dec    byte [vbe3] ; 2 = VBE2 compatible
                                                       ; (vbe3 feature is defective in this vbios)
                                    ;jmp   short display_mem_info
                                    ; 02/12/2020
00000688 E9A5010000                 jmp    display_mem_info

                                verify_pmib_chksum:
                                    ; 15/11/2020
0000068D 31C0                       xor    eax, eax
                                    ;mov   ecx, eax
                                    ;mov   cl, 20
0000068F 66B91400                   mov    cx, 20 ; 30/11/2020
00000693 56                         push   esi
                                pmib_sum_bytes:
00000694 AC                         lodsb
00000695 00C4                       add    ah, al
00000697 E2FB                       loop   pmib_sum_bytes
00000699 5E                         pop    esi
0000069A 08E4                       or     ah, ah
0000069C 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)

0000069E C605[DB860100]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
000006A5 89F7                       mov    edi, esi

                                    ;mov   esi, [VBE3INFOBLOCK+22] ; 097E00h + 16h
                                    ;            ; OemVendorNamePtr (seg16:off16)
000006A7 8B35067E0900               mov    esi, [VBE3INFOBLOCK+6] ; 097E00h + 06h
                                                ; OemStringPtr (seg16:off16)
000006AD 30C0                   xor    al, al  ; eax = 0
000006AF 6696                   xchg   ax, si ; ax = offset, si = 0
000006B1 C1EE0C                 shr    esi, 12 ; (to convert segment to base addr)
000006B4 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
000006B7 B40C                   mov    ah, 0Ch ; Black background, light red forecolor

000006B9 E8E23B0000             call   print_kmsg

                                ;mov   ah, 07h

000006BE BE[BD860100]           mov    esi, vesa_vbe3_bios_msg
                                ;call  print_kmsg
000006C3 E8DE3B0000             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 <res 00000004>          .Signature: resb 4  ; db 'PMID' ; PM Info Block Signature
00000004 <res 00000002>          .EntryPoint:      resw 1 ; Offset of PM entry point within BIOS
00000006 <res 00000002>          .PMInitialize: resw 1    ; Offset of PM initialization entry point
00000008 <res 00000002>          .BIOSDataSel:     resw 1       ; Selector to BIOS data area emulation block
0000000A <res 00000002>          .A0000Sel:  resw 1 ; Selector to access A0000h physical mem
0000000C <res 00000002>          .B0000Sel:  resw 1  ; Selector to access B0000h physical mem
0000000E <res 00000002>          .B8000Sel:  resw 1 ; Selector to access B8000h physical mem
00000010 <res 00000002>          .CodeSegSel:      resw 1 ; Selector to access code segment as data
00000012 <res 00000001>          .InProtectMode: resb 1 ; Set to 1 when in protected mode
00000013 <res 00000001>          .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
000006C8 31C0                         xor    eax, eax
000006CA B900000100                   mov    ecx, 65536
000006CF E8B15D0000                   call   allocate_memory_block
                                      ;jc    short di4
000006D4 0F8251010000                 jc     di0 ; 30/11/2020
                                
                                      ; of course this block must be in the 1st 16MB
                                      ; because vbe3 pmi segments will be 16 bit segments
                                      ; (80286 type segment descriptors in GDT)
                                
000006DA A3[20120300]                 mov    [vbe3bios_addr], eax
                                
                                      ; set [pmid_addr] to the new location
000006DF BE00000C00                   mov    esi, 0C0000h
                                
                                      ; 30/11/2020
000006E4 29F7                         sub    edi, esi ; izolate offset
000006E6 01C7                         add    edi, eax ; new address
000006E8 893D[24120300]               mov    [pmid_addr], edi ; new 'PMID' location
                                
                                      ; Move VIDEO BIOS from 0C0000h to EAX
000006EE B900400000                   mov    ecx, 65536/4
000006F3 89C7                         mov    edi, eax ; 30/11/2020
000006F5 F3A5                         rep    movsd
                                
                                      ; 02/12/2020
                                      ; 30/11/2020
                                      ; set vbe3 segment selectors
                                
                                      ; VBE3CS (VESA VBE3 video bios code segment)
000006F7 BF[326D0000]                 mov    edi, _vbe3_CS+2 ; base address bits 0..15
000006FC 66AB                         stosw  ; edi = _vbe3_CS+4
000006FE C1C810                       ror    eax, 16
00000701 8807                         mov    [edi], al ; base address, bits 16..23
                                
                                      ; VBE3DS ('CodeSegSel' in PMInfoBlock)
00000703 BF[5C6D0000]                 mov    edi, _vbe3_DS+4 ; base addr bits 16..23
00000708 8807                         mov    [edi], al
0000070A C1C010                       rol    eax, 16
0000070D 668947FE                     mov    [edi-2], ax ; base address, bits 0..15
                                
                                      ; VBE3BDS (BIOSDataSel in PMInfoBlock)
00000711 BF[3A6D0000]                 mov    edi, _vbe3_BDS+2 ; base addr bits 0..15
00000716 B800700900                   mov    eax, VBE3BIOSDATABLOCK ; 1536 bytes
0000071B 66AB                         stosw  ; edi = _vbe3_BDS+4
0000071D C1E810                       shr    eax, 16
00000720 8807                         mov    [edi], al ; base address, bits 16..23
                                
                                      ; VBE3SS (1024 bytes)
00000722 BF[626D0000]                 mov    edi, _vbe3_SS+2 ; base addr bits 0..15
00000727 B800600900                   mov    eax, VBE3STACKADDR ; size = 1024 bytes
0000072C 66AB                         stosw  ; edi = _vbe3_SS+4
0000072E C1E810                       shr    eax, 16
00000731 8807                         mov    [edi], al ; base address, bits 16..23
                                
                                      ; stack pointer (esp) will be set to 1020
                                      ; (before VBE3 PMI call)
                                
                                      ; VBE3ES (max: 2048 bytes)
00000733 BF[6A6D0000]                 mov    edi, _vbe3_ES+2 ; base addr bits 0..15
00000738 B800760900                   mov    eax, VBE3SAVERESTOREBLOCK
0000073D 66AB                         stosw  ; edi = _vbe3_ES+4
0000073F C1E810                       shr    eax, 16
00000742 8807                         mov    [edi], al ; base address, bits 16..23
                                
                                      ;Note: low word of _VBE3_ES base address will be
                                      ;      set -again- by VBE3 PMI caller routine
                                
                                      ; 09/12/2020
                                      ;; set pmi32 (as VBE3 PMI is ready)
                                      ;inc  byte [pmi32] ; = 1
                                
                                      ; KCODE16 (set PMI far return segment)
00000744 BF[726D0000]                 mov    edi, _16bit_CS+2 ; base addr bits 0..15
00000749 B8[D2070000]                 mov    eax, pminit_return_addr16
0000074E 66AB                         stosw  ; edi = _16bit_CS+4
00000750 C1E810                       shr    eax, 16
00000753 8807                         mov    [edi], al ; base address, bits 16..23
                                
                                      ; 30/11/2020
                                      ; clear mem from VBE3 BIOS data area emu block
                                      ; to end of vbe3 buffers
```

```
                                        ; 01/12/2020
00000755 BF00700900                     mov   edi, VBE3BIOSDATABLOCK ; 97000h
                                        ;mov  cx, (VBE3INFOBLOCK-VBE3BIOSDATABLOCK)/4
                                        ;     ; ecx = 3584/4 double words
                                        ; 21/12/2020
0000075A 66B90003                       mov   cx, (VBE3MODEINFOBLOCK-VBE3BIOSDATABLOCK)/4
                                              ; ecx = 3072/4 double words
                                        ;xor  eax, eax
0000075E 30C0                           xor   al, al
00000760 F3AB                           rep   stosd

                                        ; Filling PMInfoBlock selector fields
00000762 8B3D[24120300]                 mov   edi, [pmid_addr]
00000768 66C747083800                   mov   word [edi+PMInfo.BIOSDataSel], VBE3BDS
0000076E 66C7470A4000                   mov   word [edi+PMInfo.A0000Sel], VBE3A000
00000774 66C7470C4800                   mov   word [edi+PMInfo.B0000Sel], VBE3B000
0000077A 66C7470E5000                   mov   word [edi+PMInfo.B8000Sel], VBE3B800
00000780 66C747105800                   mov   word [edi+PMInfo.CodeSegSel], VBE3DS
00000786 C6471201                       mov   byte [edi+PMInfo.InProtectMode], 1

                                        ; Calculate and write checksum byte
0000078A 89FE                           mov   esi, edi
0000078C B113                           mov   cl, PMInfo.size - 1
                                        ;xor  ah, ah
                                pmid_chksum:
0000078E AC                             lodsb
0000078F 00C4                           add   ah, al
00000791 E2FB                           loop  pmid_chksum
00000793 F6DC                           neg   ah ; 1 -> 255, 255 -> 1
00000795 8826                           mov   byte [esi], ah  ; checksum

                                        ; far call PM initialization
                                        ; (VBE3 video bios will return via 'retf')

00000797 668B4706                       mov   ax, [edi+PMInfo.PMInitialize]
                                        ; 30/11/2020
0000079B C1E010                         shl   eax, 16 ; save entry address in hw
                                        ; ax = 0

                                        ; 02/12/2020
0000079E 68[F6070000]                   push  pminit_ok ; normal, near return address

                                        ; 30/11/2020
                                _VBE3PMI_fcall:
                                        ; ax = function, hw of eax = entry address
000007A3 9C                             pushf ; save 32 bit flags
000007A4 56                             push  esi ; *
000007A5 55                             push  ebp ; **

000007A6 89E5                           mov   ebp, esp ; save 32 bit stack pointer

000007A8 89C6                           mov   esi, eax

000007AA FA                             cli

                                        ; Disable interrupts (clear interrupt flag)
                                        ; Reset Interrupt MASK Registers (Master&Slave)
000007AB B0FF                           mov   al, 0FFh     ; mask off all interrupts
000007AD E621                           out   21h, al            ; on master PIC (8259)
000007AF EB00                           jmp   $+2  ; (delay)
000007B1 E6A1                           out   0A1h, al     ; on slave PIC (8259)

                                        ; 02/12/2020
000007B3 66B86000                       mov   ax, VBE3SS
000007B7 8ED0                           mov   ss, ax

000007B9 BCFC030000                     mov   esp, 1020 ; 30/11/2020

                                        ; 01/12/2020
                                        ;lss  esp, [stack16]

000007BE C1E810                         shr   eax, 16      ; now, entry address is in lw

                                        ; 30/11/2020 - 16 bit pm selector test (OK)
                                        ; (32 bit stack push/pop & retf with 32 bit code segment)
                                        ; (16 bit stack push/pop with 16 bit code segment)

                                        ; return
                                        ;push  KCODE16
                                        ;push  0 ; 30/11/2020 (pminit_return_addr16)

                                        ; 30/11/2020 (16 bit stack during retf from video bios)
000007C1 C7042400007000                 mov   dword [esp], KCODE16 << 16
                                                           ; ip = 0, cs = KCODE16
                                        ; 01/12/2020
                                        ;mov  dword [VBE3STACKADDR+1020], KCODE16*65536

                                        ;mov  [jumpfar16], eax

                                        ; 02/12/2020
                                        ; 30/11/2020 (32 bit stack during retf from kernel)
                                        ; far jump/call via retf
000007C8 6A30                           push  VBE3CS ; VBE3 video bios's code segment
000007CA 50                             push  eax ; PMInitialize or EntryPoint

000007CB 6689F0                         mov   ax, si ; restore function

                                        ; 02/12/2020
000007CE 31F6                           xor   esi, esi ; (not necessary, it is not used)

000007D0 CB                             retf  ; far return (to 16 bit code segment)

                                        ; 01/12/2020
                                        ;db   0EAh  ; far jump to 16 bit code segment
```

```
                                ;jumpfar16:
                                      ;dd    0
                                      ;dw    VBE3CS
                                ;stack16:
                                      ;dd    1020
                                      ;dw    VBE3SS
000007D1 90                           align 2
                                pminit_return_addr16:
                                      ; 02/12/2020
                                      ; 30/11/2020
                                      ;;db   66h              ; Prefix for 32-bit
                                      ;db    0EAh             ; Opcode for far jump
                                      ;dd    pminit_return_addr32 ; 32 bit Offset
                                      ;dw    KCODE            ; 32 bit code segment
                                      ; 01/12/2020
000007D2 EA[D9070000]0800             jmp    KCODE:pminit_return_addr32
                                pminit_return_addr32:
                                      ; restore 32 bit kernel selectors and 32 bit stack addr
000007D9 BE10000000                   mov    esi, KDATA
000007DE 8EDE                         mov    ds, si
000007E0 8EC6                         mov    es, si
000007E2 8ED6                         mov    ss, si
000007E4 89EC                         mov    esp, ebp  ; top of stack = iretd return addr
000007E6 5D                           pop    ebp ; **
000007E7 5E                           pop    esi ; *
000007E8 9D                           popf   ; restore 32 bit flags
                                      ; enable interrupts
000007E9 FA                           cli
                                      ; 21/12/2020
000007EA 50                           push   eax
000007EB 30C0                         xor    al, al    ; Enable all hardware interrupts!
000007ED E621                         out    21h, al         ; (IBM PC-AT compatibility)
000007EF EB00                         jmp    $+2       ; (All conventional PC-AT hardware
000007F1 E6A1                         out    0A1h, al   ;  interrupts will be in use.)
                                                       ; (Even if related hardware component
                                                       ;  does not exist!)
000007F3 58                           pop    eax
000007F4 FB                           sti
                                      ; top of stack = return address
                                      ; ('pminit_ok' for PMinit)
000007F5 C3                           retn
                                pminit_ok:
                                      ; 03/12/2020
                                      ; (set [pmid_addr] to PMI entry point for next calls)
000007F6 8305[24120300]04             add    dword [pmid_addr], PMInfo.EntryPoint ; + 4
                                      ; 17/01/2021
                                      ; copy EDID data from temporary location to final address
000007FD 803D[37430000]4F             cmp    byte [edid], 4Fh
00000804 7511                         jne    short vbe3h_chcl
00000806 B920000000                   mov    ecx, 32 ; 128 bytes, 32 dwords
0000080B BE007D0900                   mov    esi, VBE3EDIDINFOBLOCK ; 97D00h
00000810 BF[9C110300]                 mov    edi, edid_info
00000815 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
00000817 BE40810B00                   mov    esi, 0B8000h + 160*2 ; row 2
0000081C 89F7                         mov    edi, esi
                                vbe3h_chcl_next:
0000081E 66AD                         lodsw
```

```
1340 00000820 80FC0C                    cmp    ah, 0Ch      ; light red forecolor
1341 00000823 750D                      jne    short display_mem_info
1342 00000825 B40E                      mov    ah, 0Eh ; yellow forecolor
1343 00000827 66AB                      stosw
1344 00000829 EBF3                      jmp    short vbe3h_chcl_next
1345
1346                                 di0:
1347                                    ; 30/11/2020
1348                                    ; Memory allocation error !
1349 0000082B C605[5C090000]00          mov    byte [vbe3], 0 ; disable VBE3
1350
1351                                 display_mem_info:
1352                                    ; 19/12/2020
1353                                    ; temporary
1354 00000832 E8853A0000                call   default_lfb_info
1355                                    ;
1356                                    ; 06/11/2014
1357 00000837 E80B3A0000                call   memory_info
1358                                    ; 14/08/2015
1359                                    ;call getch ; 28/02/2015
1360                                 drv_init:
1361 0000083C FB                        sti    ; Enable Interrupts
1362                                    ; 06/02/2015
1363 0000083D 8B15[106E0000]            mov    edx, [hd0_type] ; hd0, hd1, hd2, hd3
1364 00000843 668B1D[0E6E0000]          mov    bx, [fd0_type] ; fd0, fd1
1365                                    ; 22/02/2015
1366 0000084A 6621DB                    and    bx, bx
1367 0000084D 756C                      jnz    short di1
1368                                    ;
1369 0000084F 09D2                      or     edx, edx
1370 00000851 757A                      jnz    short di2
1371                                    ;
1372                                 setup_error:
1373 00000853 BE[2A4D0100]              mov    esi, setup_error_msg
1374                                 psem:
1375 00000858 AC                        lodsb
1376 00000859 08C0                      or     al, al
1377                                    ;jz    short haltx ; 22/02/2015
1378 0000085B 747B                      jz     short di3
1379 0000085D 56                        push   esi
1380                                    ; 13/05/2016
1381 0000085E BB07000000                mov    ebx, 7 ; Black background,
1382                                                  ; light gray forecolor
1383                                                  ; Video page 0 (BH=0)
1384 00000863 E89D1A0000                call   _write_tty
1385 00000868 5E                        pop    esi
1386 00000869 EBED                      jmp    short psem
1387
1388                                 check_boch_plex86_vbe:
1389                                    ; 20/11/2020
1390                                    ; check Bochs/Plex86 VGABios VBE extension
1391                                    ; (check if TRDOS 386 v2 is running on emulators)
1392                                    ; BOCHS/QEMU/VIRTUALBOX
1393                                    ;
1394                                    ; ref: vbe_display_api.txt
1395
1396                                    ; bochs/plex86 VGAbios VBE source code
1397                                    ;  by Jeroen Janssen (2002)
1398                                    ;  and Volker Ruppert (2003-2020)
1399
1400 0000086B 29C0                      sub    eax, eax ; 0
1401 0000086D 66BACE01                  mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
1402 00000871 66EF                      out    dx, ax ; VBE_DISPI_INDEX_ID register
1403                                    ;mov   ax, 0B0C0h ; VBE_DISPI_ID0
1404                                    ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
1405 00000873 6642                      inc    dx
1406                                    ;out   dx, ax
1407                                    ;nop
1408 00000875 66ED                      in     ax, dx
1409 00000877 80FCB0                    cmp    ah, 0B0h
1410                                    ;jne   short not_boch_qemu_vbe
1411 0000087A 75B6                      jne    short display_mem_info
1412 0000087C 3CC5                      cmp    al, 0C5h ; it must be 0B0C4h or 0B0C5h ..
1413                                    ;ja    short not_boch_qemu_vbe
1414 0000087E 77B2                      ja     short display_mem_info
1415 00000880 3CC0                      cmp    al, 0C0h ; 0B0C0h to 0B0C5h .. ; Qemu
1416                                    ;cmp   al, 0C4h ; 0BC04h or 0B0C5h is OK ; Bochs
1417                                    ;jb    short not_boch_qemu_vbe
1418 00000882 72AE                      jb     short display_mem_info
1419
1420                                    ; save VESA VBE2 bios (bochs/qemu) signature
1421                                    ; for enabling VBE2 functions in TRDOS 386 v2 kernel
1422 00000884 A2[5D090000]              mov    [vbe2bios], al ; 0C4h or 0C5h (for BOCHS)
1423                                                          ; (0C0h-0C5h for QEMU)
1424 00000889 C605[C6860100]32          mov    byte [vbe_vnumber], "2"
1425                                    ; 26/11/2020
1426                                    ; "BOCHS/QEMU/VIRTUALBOX VBE2 Video BIOS ..".
1427 00000890 BE[A8860100]              mov    esi, vbe2_bochs_vbios ; BOCH/QEMU vbios msg
1428 00000895 B40E                      mov    ah, 0Eh  ; Yellow font
1429 00000897 E8043A0000                call   print_kmsg
1430
1431                                    ; this is not necessary ! (20/11/2020)
1432 0000089C 803D[5D090000]C4          cmp    byte [vbe2bios], 0C4h
1433 000008A3 728D                      jb     display_mem_info    ; (QEMU)
1434
1435                                    ; Display kernel version message if 0E9h hack port
1436                                    ; is enabled (bochs emulator feature)
1437 000008A5 66BAE900                  mov    dx, 0E9h ; hack port for BOCHS
1438 000008A9 BE[01870100]              mov    esi, kernel_version_msg
1439                                 kvmsg_next_char:
1440 000008AE AC                        lodsb
1441 000008AF 08C0                      or     al, al
1442 000008B1 7505                      jnz    short put_kvmsg_in_hack_port
1443                                 not_boch_qemu_vbe:
1444 000008B3 E97AFFFFFF                jmp    display_mem_info
```

```
                                    put_kvmsg_in_hack_port:
000008B8 EE                                 out    dx, al
000008B9 EBF3                               jmp    short kvmsg_next_char

                                    di1:
                                            ; supress 'jmp short T6'
                                            ;  (activate fdc motor control code)
000008BB 66C705[BE090000]90-                mov    word [T5], 9090h ; nop
000008C3 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
                                            ;
000008C4 E82A490000                         CALL   DSKETTE_SETUP; Initialize Floppy Disks
                                            ;
000008C9 09D2                               or     edx, edx
000008CB 740B                                 jz      short di3
                                    di2:
000008CD E868490000                         call         DISK_SETUP   ; Initialize Fixed Disks
000008D2 0F827BFFFFFF                         jc      setup_error
                                    di3:
000008D8 E83E390000                         call   setup_rtc_int; 22/05/2015 (dsectrpm.s)
                                            ;
000008DD E8DE420100                         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

000008E2 803D[5C090000]02                   cmp    byte [vbe3], 2
000008E9 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

000008EB BEFE7B0900                         mov    esi, VBE3MODEINFOBLOCK - 2
000008F0 66C7061801                         mov    word [esi], 0118h ; default vbe mode
                                                                      ; 1024*768, 24bpp
000008F5 E8AD310000                         call   set_lfbinfo_table

000008FA E8DC600000                         call   allocate_lfb_pages_for_kernel
                                    di4:
                                            ; 08/09/2016
000008FF 0F20C0                             mov    eax, cr0
00000902 A810                               test   al, 10h  ; Bit 4, ET (Extension Type)
00000904 7408                               jz     short sysinit
                                            ; 27/02/2017
00000906 FE05[D4960100]                     inc    byte [fpready]
                                            ; 80387 (FPU) is ready
0000090C DBE3                               fninit ; Initialize Floating-Point Unit
                                    sysinit:
                                            ; 30/06/2015
0000090E E8336B0000                         call   sys_init
                                            ;
                                            ;jmp   cpu_reset ; 22/02/2015
                                    hang:
                                            ; 23/02/2015
                                            ;sti                ; Enable interrupts
00000913 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
00000914 31DB                               xor    ebx, ebx
```

```
1549 00000916 8A1D[468A0100]            mov    bl, [ptty]    ; active_page
1550 0000091C 89DE                      mov    esi, ebx
1551 0000091E 66D1E6                    shl    si, 1
1552 00000921 81C6[488A0100]            add    esi, ttychr
1553 00000927 668B06                    mov    ax, [esi]
1554 0000092A 6621C0                    and    ax, ax
1555                                    ;jz    short _c8
1556 0000092D 74E4                      jz     short hang
1557 0000092F 66C7060000                mov    word [esi], 0
1558 00000934 80FB03                    cmp    bl, 3         ; Video page 3
1559                                    ;jb    short _c8
1560 00000937 72DA                      jb     short hang
1561                                    ;
1562                                    ; 13/05/2016
1563                                    ; 07/09/2014
1564                            nxtl:
1565 00000939 6653                      push   bx
1566 0000093B 66BB0E00                  mov    bx, 0Eh       ; Yellow character
1567                                                          ; on black background
1568                                                          ; bh = 0 (video page 0)
1569                                                          ; Retro UNIX 386 v1 - Video Mode 0
1570                                                          ; (PC/AT Video Mode 3 - 80x25 Alpha.)
1571 0000093F 6650                      push   ax
1572 00000941 E8BF190000                call   _write_tty
1573 00000946 6658                      pop    ax
1574 00000948 665B                      pop    bx
1575 0000094A 3C0D                      cmp    al, 0Dh              ; carriage return (enter)
1576                                    ;jne   short _c8
1577 0000094C 75C5                      jne    short hang
1578 0000094E B00A                      mov    al, 0Ah              ; next line
1579 00000950 EBE7                      jmp    short nxtl
1580
1581                            ; c8:
1582                            ;       ; 25/08/2014
1583                            ;       cli                         ; Disable interrupts
1584                            ;       mov    al, [scounter + 1]
1585                            ;       and    al, al
1586                            ;       jnz    hang
1587                            ;       call   rtc_p
1588                            ;       jmp     hang
1589
1590                                    ; 27/08/2014
1591                                    ; 20/08/2014
1592                            printk:
1593                                      ;mov    edi, [scr_row]
1594                            pkl:
1595 00000952 AC                        lodsb
1596 00000953 08C0                      or     al, al
1597 00000955 7404                      jz     short pkr
1598 00000957 66AB                      stosw
1599 00000959 EBF7                      jmp    short pkl
1600                            pkr:
1601 0000095B C3                        retn
1602
1603                            ; 14/11/2020 (TRDOS 386 v2.0.3)
1604 0000095C 00                vbe3: db 0  ; VESA VBE version (must be 03h)
1605                                        ; for using video bios calls in protected mode
1606                            vbe2bios:
1607 0000095D B0                        db 0B0h ;
1608                            ;pmid_addr:
1609                                    ;dw 0  ; > 0 if 'PMID' sign is found
1610                                    ;       ;       ('pmid' offset addr in VGA bios seg, 0C000h)
1611                                    ;; 02/12/2020
1612                                    ;dw 0  ; 32 bit address in pmid_addr
1613
1614                            ; 28/02/2017
1615                            ; 22/01/2017
1616                            ; 15/01/2017
1617                            ; 14/01/2017
1618                            ; 02/01/2017
1619                            ; 25/12/2016
1620                            ; 19/12/2016
1621                            ; 10/12/2016 (callback)
1622                            ; 06/06/2016
1623                            ; 23/05/2016
1624                            ; 22/05/2016 - TRDOS 386 (TRDOS v2.0) Timer Event Modifications
1625                            ; 25/07/2015
1626                            ; 14/05/2015 (multi tasking -time sharing- 'clock', x_timer)
1627                            ; 17/02/2015
1628                            ; 06/02/2015 (unix386.s)
1629                            ; 11/12/2014 - 22/12/2014 (dsectrm2.s)
1630                            ;
1631                            ; IBM PC-XT Model 286 Source Code - BIOS2.ASM (06/10/85)
1632                            ;
1633                            ;-- HARDWARE INT  08 H - ( IRQ LEVEL 0 ) ----------------------------------------
1634                            ;     THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM FROM CHANNEL 0 OF        :
1635                            ;     THE 8254 TIMER.  INPUT FREQUENCY IS 1.19318 MHZ AND THE DIVISOR        :
1636                            ;     IS 65536, RESULTING IN APPROXIMATELY 18.2 INTERRUPTS EVERY SECOND.     :
1637                            ;                                                                            :
1638                            ;     THE INTERRUPT HANDLER MAINTAINS A COUNT (40:6C) OF INTERRUPTS SINCE    :
1639                            ;     POWER ON TIME, WHICH MAY BE USED TO ESTABLISH TIME OF DAY.        :
1640                            ;     THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR CONTROL COUNT (40:40)  :
1641                            ;     OF THE DISKETTE, AND WHEN IT EXPIRES, WILL TURN OFF THE           :
1642                            ;     DISKETTE MOTOR(s), AND RESET THE MOTOR RUNNING FLAGS.                    :
1643                            ;     THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE THROUGH            :
1644                            ;     INTERRUPT 1CH AT EVERY TIME TICK.  THE USER MUST CODE A           :
1645                            ;     ROUTINE AND PLACE THE CORRECT ADDRESS IN THE VECTOR TABLE.        :
1646                            ;-------------------------------------------------------------------------------
1647                            ;
1648
1649                            timer_int:  ; IRQ 0
1650                            ;int_08h:   ; Timer
1651                                  ; 14/10/2015
1652                                  ; Here, we are simulating system call entry (for task switch)
1653                                  ; (If multitasking is enabled,
```

```
                                        ; 'clock' procedure may jump to 'sysrelease')
                                        push   ds
0000095E 1E                             push   es
0000095F 06                             push   fs
00000960 0FA0                           push   gs
00000962 0FA8
                                        pushad ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
00000964 60                             mov     cx, KDATA
00000965 66B91000                         mov     ds, cx
00000969 8ED9                             mov     es, cx
0000096B 8EC1                             mov     fs, cx
0000096D 8EE1                             mov     gs, cx
0000096F 8EE9
                                        mov    ecx, cr3
00000971 0F20D9                         mov    [cr3reg], ecx ; save current cr3 register value/content
00000974 890D[5C040300]
                                        ; 14/01/2017
                                        cmp    ecx, [k_page_dir]
0000097A 3B0D[188A0100]                 je     short T3
00000980 7409
                                        mov    ecx, [k_page_dir]
00000982 8B0D[188A0100]                 mov    cr3, ecx
00000988 0F22D9                 T3:
                                        ;sti                    ; INTERRUPTS BACK ON
                                        INC    word [TIMER_LOW] ; INCREMENT TIME
0000098B 66FF05[988A0100]               JNZ    short T4         ; GO TO TEST_DAY
00000992 7507                           INC    word [TIMER_HIGH]; INCREMENT HIGH WORD OF TIME
00000994 66FF05[9A8A0100]       T4:                             ; TEST_DAY
                                        CMP    word [TIMER_HIGH],018H    ; TEST FOR COUNT EQUALING 24 HOURS
0000099B 66833D[9A8A0100]18             JNZ    short T5         ; GO TO DISKETTE_CTL
000009A3 7519                           CMP    word [TIMER_LOW],0B0H
000009A5 66813D[988A0100]B0-
000009AD 00
                                        JNZ    short T5         ; GO TO DISKETTE_CTL
000009AE 750E
                                ;-----  TIMER HAS GONE 24 HOURS
                                        ;;SUB  AX,AX
                                        ;MOV   [TIMER_HIGH],AX
                                        ;MOV   [TIMER_LOW],AX
                                        sub    eax, eax
000009B0 29C0                           mov    [TIMER_LH], eax
000009B2 A3[988A0100]                   ;
                                        MOV    byte [TIMER_OFL],1
000009B7 C605[9C8A0100]01
                                ;-----  TEST FOR DISKETTE TIME OUT
                                T5:
                                        ; 23/12/2014
                                        jmp    short T6        ; will be replaced with nop, nop
                                                                ; (9090h) if a floppy disk
                                                                ; is detected.
                                        ;mov   al,[CS:MOTOR_COUNT]
000009C0 A0[9F8A0100]                   mov    al, [MOTOR_COUNT]
000009C5 FEC8                           dec    al
                                        ;mov   [CS:MOTOR_COUNT], al      ; DECREMENT DISKETTE MOTOR CONTROL
000009C7 A2[9F8A0100]                   mov    [MOTOR_COUNT], al
                                        ;mov   [ORG_MOTOR_COUNT], al
000009CC 750F                           JNZ    short T6        ; RETURN IF COUNT NOT OUT
000009CE B0F0                           mov    al,0F0h
                                        ;AND   [CS:MOTOR_STATUS],al      ; TURN OFF MOTOR RUNNING BITS
000009D0 2005[9E8A0100]                 and    [MOTOR_STATUS], al
                                        ;and   [ORG_MOTOR_STATUS], al
000009D6 B00C                           MOV    AL,0CH          ; bit 3 = enable IRQ & DMA,
                                                               ; bit 2 = enable controller
                                                               ;      1 = normal operation
                                                               ;      0 = reset
                                                               ; bit 0, 1 = drive select
                                                               ; bit 4-7 = motor running bits
000009D8 66BAF203                       MOV    DX,03F2H        ; FDC CTL PORT
000009DC EE                             OUT    DX,AL           ; TURN OFF THE MOTOR
                                T6:
                                        ;inc   word [CS:wait_count]      ; 22/12/2014 (byte -> word)
                                                               ; TIMER TICK INTERRUPT
                                    ;;inc  word [wait_count] ;;27/02/2015
                                        ;INT   1CH              ; TRANSFER CONTROL TO A USER ROUTINE
                                        ;cli
000009DD E857040000                     call   u_timer                  ; TRANSFER CONTROL TO A USER ROUTINE
                                        ; 23/05/2016
000009E2 E818230100                     call   clock            ; Multi Tasking control procedure
                                T7:
                                        ; 14/10/2015
000009E7 B020                           MOV    AL,EOI           ; GET END OF INTERRUPT MASK
000009E9 FA                             CLI                     ; DISABLE INTERRUPTS TILL STACK CLEARED
000009EA E620                           OUT    INTA00,AL        ; END OF INTERRUPT TO 8259 - 1
                                        ;
                                rtc_int_2:
                                        ; 26/12/2016
                                        ;mov   ecx, [cr3reg]
                                        ; 13/01/2017
000009EC 803D[D4030300]00               cmp    byte [u.t_lock], 0       ; T_LOCK
000009F3 7730                           ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                        ; 28/02/2017
                                        ; We need to exit if the user's IRQ callback service is in progress!
                                        ; (To prevent a conflict!)
000009F5 803D[D8030300]00               cmp    byte [u.r_lock], 0  ; R_LOCK, IRQ callback service lock !
000009FC 7727                           ja     short timer_int_return  ; Timer Lock : 'sysrele' is needed !
                                        ; 15/01/2017
000009FE 803D[A8960100]02               cmp    byte [priority], 2
00000A05 733A                           jnb    short T8  ; current process has a timer event (15/01/2017)
                                        ; 22/05/2016
00000A07 803D[A9960100]00               cmp    byte [p_change], 0 ; in 'set_run_sequence', in 'rtc_p'
00000A0E 7615                           jna    short timer_int_return ; 23/05/2016

                                        ; 15/01/2017
```

```
1758                                    ; present process must be changed with high priority process
1759                                    ;xor   al, al
1760 00000A10 31C0                     xor    eax, eax ; 26/12/2016
1761 00000A12 A2[A9960100]              mov    [p_change], al ; 0
1762                                    ;mov   byte [priority], 2 ; 15/01/2017 (there is a timer event)
1763
1764 00000A17 803D[5B030300]FF          cmp     byte [sysflg], 0FFh ; user or system space ?
1765 00000A1E 7416                      je     short rtc_int_3     ; user space ([sysflg]= 0FFh)
1766
1767                                    ; system space, wait for 'sysret'
1768                                    ; to change running process
1769                                    ; with high priority (event) process
1770
1771 00000A20 A2[A8030300]              mov    [u.quant], al ; 0
1772
1773                                timer_int_return: ; 23/05/2016 - jump from 'rtc_int' ('rtc_int_2')
1774 00000A25 8B0D[5C040300]            mov    ecx, [cr3reg]       ; previous value/content of cr3 register
1775 00000A2B 0F22D9                    mov    cr3, ecx     ; restore cr3 register content
1776                                    ;
1777 00000A2E 61                        popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
1778                                    ;
1779 00000A2F 0FA9                      pop    gs
1780 00000A31 0FA1                      pop    fs
1781 00000A33 07                        pop    es
1782 00000A34 1F                        pop    ds
1783                                    ;
1784 00000A35 CF                        iretd  ; return from interrupt
1785
1786                                rtc_int_3:
1787 00000A36 FE05[5B030300]            inc    byte [sysflg]       ; now, we are in system space
1788                                    ;
1789 00000A3C E9EBCE0000                  jmp     sysrelease ; change running process immediatelly
1790
1791                                T8:
1792                                    ; 13/01/2017 (eax -> ebx)
1793                                    ; callback checking... (19/12/2016)
1794 00000A41 31DB                      xor    ebx, ebx
1795 00000A43 871D[D0030300]            xchg   ebx, [u.tcb] ; callback address (0 = normal return)
1796 00000A49 09DB                      or     ebx, ebx
1797 00000A4B 74D8                      jz     short timer_int_return
1798
1799                                    ; Set user's callback routine as return address from this interrupt
1800                                    ; and set normal return address as return address from callback
1801                                    ; routine!!! (19/12/2016)
1802
1803                                    ; 14/01/2017
1804                                    ; 13/01/2017 - Timer Lock (T_LOCK)
1805 00000A4D FE05[D4030300]            inc    byte [u.t_lock]
1806 00000A53 8A0D[5B030300]            mov    cl, [sysflg]
1807 00000A59 880D[D5030300]            mov    [u.t_mode], cl
1808
1809 00000A5F 8B2D[B4890100]            mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
1810 00000A65 83ED14                    sub    ebp, 20             ; eip, cs, eflags, esp, ss
1811 00000A68 892D[5C030300]            mov    [u.sp], ebp
1812 00000A6E 8925[60030300]            mov    [u.usp], esp
1813
1814                                    ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
1815
1816 00000A74 8B44241C                  mov    eax, [esp+28] ; pushed eax
1817 00000A78 A3[64030300]              mov    [u.r0], eax
1818
1819 00000A7D E8790F0100                call   wswap ; save user's registers & status
1820
1821                                    ; software int is in ring 0 but timer int must return to ring 3
1822                                    ; so, ring 3 return address and stack registers
1823                                    ; (eip, cs, eflags, esp, ss)
1824                                    ; must be copied to timer int return
1825                                    ; eip will be replaced by callback service routine address
1826
1827 00000A82 C605[5B030300]FF          mov    byte [sysflg], 0FFh ; user mode
1828
1829                                    ; system mode (system call)
1830                                    ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
1831                                                    ; ESP (u), SS (UDATA)
1832
1833 00000A89 8B4510                    mov    eax, [ebp+16]; SS (UDATA
1834 00000A8C 89E6                      mov    esi, esp
1835 00000A8E 50                        push   eax
1836 00000A8F 50                        push   eax
1837 00000A90 89E7                      mov    edi, esp
1838 00000A92 893D[60030300]            mov    [u.usp], edi
1839 00000A98 B908000000                mov    ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
1840 00000A9D F3A5                      rep    movsd
1841 00000A9F B104                      mov    cl, 4
1842 00000AA1 F3AB                      rep    stosd
1843 00000AA3 893D[5C030300]            mov    [u.sp], edi
1844 00000AA9 89EE                      mov    esi, ebp
1845 00000AAB B105                      mov    cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
1846 00000AAD F3A5                      rep    movsd
1847
1848 00000AAF 8B0D[B8030300]            mov    ecx, [u.pgdir]
1849 00000AB5 890D[5C040300]            mov    [cr3reg], ecx
1850
1851                                    ; 13/01/207 (eax -> ebx)
1852                                    ; EBX = callback routine address (virtual, not physical address!)
1853
1854                                    ; 09/01/2017
1855                                    ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'sysrele'
1856                                    ;     system call !!!
1857                                    ; 25/12/2016
1858                                    ; Callback Note: (19/12/2016)
1859                                    ; !!! CALLBACK ROUTINE MUST BE ENDED/RETURNED WITH 'RETN' !!!
1860                                    ;     pushf ; save flags
1861                                    ;     <callback service code>
1862                                    ;     popf  ; restore flags
```

```
                                ;      retn ; return to normal running address
                                ;
                                ; 15/01/2017
                                ; 14/01/2017
                                ; 13/01/2017 (eax -> ebx)
                                ; 10/01/2017
                              set_callback_addr:
                                ; 09/01/2017 (**)
                                ; 02/01/2017 (*)
                                ; 25/12/2016 (*)
                                ; 19/12/2016 (TRDOS 386 feature only!)
                                ;
                                ; This routine sets return address
                                ; to start of user's interrupt
                                ; service (callback) address
                                ;; and sets callback 'retn' address to normal
                                ;; return address of user's running code!
                                ;
                                ; INPUT:
                                ;      EBX = callback routine/service address
                                ;            (virtual, not physical address!)
                                ;      [u.sp] = kernel stack, points to
                                ;               user's EIP,CS,EFLAGS,ESP,SS
                                ;               registers.
                                ; OUTPUT:
                                ;      EIP (user) = callback (service) address
                                ;      CS (user) = UCODE
                                ;      EFLAGS (user) = flags before callback
                                ;        ESP (user) = ESP-4 (user, before callback)
                                ;      [ESP](user) = EIP (user) before callback
                                ;
                                ; Note: If CPU was in user mode while entering
                                ;      the timer interrupt service routine,
                                ;      'IRET' will get return to callback routine
                                ;      immediately. If CPU was in system/kernel mode
                                ;      'iret' will get return to system call and
                                ;      then, callback routine will be return address
                                ;      from system call. (User's callback/service code
                                ;      will be able to return to normal return address
                                ;      via an 'retn' at the end.)
                                ;
                                ; Note(**): User's callback service code must be ended
                                ;      with a 'sysrele' sytstem call ! (09/01/2017)
                                ;
                                ;      For example:
                                ;
                                ;      timer_callback:
                                ;          ...
                                ;          inc      dword [time_counter]
                                ;          ...
                                ;          mov eax, 39 ; 'sysrele'
                                ;          int 40h ; TRDOS 386 system call (interrupt)
                                ;
                                ;
                                ;; Note(*): User's callback service code must preserve cpu
                                ;;     flags if it has any instructions which changes
                                ;;     flags in the service code. (25/12/2016)
                                ;;
                                ;;     For example:
                                ;;
                                ;;     timer_callback:
                                ;;         pushf ; save flags
                                ;;         ; this instruction changes zero flag
                                ;;         inc      dword [time_counter]
                                ;;         popf ; restore flags
                                ;;         retn ; return to normal user code
                                ;;              (which is interrupted by the
                                ;;               timer interput)
                                ;;
                                ; 15/01/2017
00000ABB 8B2D[5C030300]         mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
00000AC1 895D00                 mov    [ebp], ebx
00000AC4 E95CFFFFFF             jmp    timer_int_return
                                ; 15/01/2017
                                ; 13/01/2017
                                ; 19/12/2016
                                ; 06/06/2016
                                ; 23/05/2016
                                ; 22/05/2016
                                ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                ; 26/02/2015
                                ; 07/09/2014
                                ; 25/08/2014
                              rtc_int:       ; Real Time Clock Interrupt (IRQ 8)
                                ; 22/05/2016
00000AC9 1E                     push   ds ; ** ; 23/05/2016
00000ACA 50                     push   eax ; *
00000ACB 66B81000               mov    ax, KDATA
00000ACF 8ED8                   mov    ds, ax
                                ;
00000AD1 8A25[968A0100]         mov    ah, [RTC_2Hz] ;  2 Hz interrupt to 1 Hz function
00000AD7 80F401                 xor    ah, 1
00000ADA 8825[968A0100]         mov    [RTC_2Hz], ah ; 1 = 0.5 second, 0 = 1 second
00000AE0 753B                   jnz    short rtc_int_return ; half second
                                ; 1 second
                              rtc_int_0:
                                ; 22/05/2016
00000AE2 58                     pop    eax ; *
                                ;
                                ; 14/10/2015 ('timer_int')
                                ; Here, we are simulating system call entry (for task switch)
                                ; (If multitasking is enabled,
```

```
1968                                    ; 'clock' procedure may jump to 'sysrelease')
1969                                    ;push  ds ; ** ; 23/05/2016
1970 00000AE3 06                        push   es
1971 00000AE4 0FA0                      push   fs
1972 00000AE6 0FA8                      push   gs
1973 00000AE8 60                        pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
1974 00000AE9 66B91000                  mov     cx, KDATA
1975                                      ;mov    ds, cx ; 06/06/2016
1976 00000AED 8EC1                        mov     es, cx
1977 00000AEF 8EE1                        mov     fs, cx
1978 00000AF1 8EE9                        mov     gs, cx
1979                                    ;
1980 00000AF3 0F20D9                    mov    ecx, cr3
1981 00000AF6 890D[5C040300]            mov    [cr3reg], ecx ; save current cr3 register value/content
1982                                    ;
1983 00000AFC 803D[D4030300]00          cmp    byte [u.t_lock], 0 ; timer lock (callback) status ?
1984 00000B03 7711                      ja     short rtc_int_1        ; yes
1985
1986                                    ; 15/01/2017
1987 00000B05 3B0D[188A0100]            cmp    ecx, [k_page_dir]
1988 00000B0B 7409                      je     short rtc_int_1
1989
1990 00000B0D 8B0D[188A0100]            mov    ecx, [k_page_dir]
1991 00000B13 0F22D9                    mov    cr3, ecx
1992                                rtc_int_1:
1993                                    ; Timer event (kernel) functions must be performed with
1994                                    ; 1 second intervals - TRDOS 386 (TRDOS v2.0) feature ! -
1995                                    ;
1996                                    ; 25/08/2014
1997 00000B16 E81A030000                call   rtc_p  ; 19/05/2016 - major modification
1998
1999                                    ; 23/05/2016
2000 00000B1B 28E4                      sub    ah, ah ; 0
2001                                    ; 22/05/2016 - TRDOS 386 timer event modifications
2002                                rtc_int_return: ; 19/05/2016
2003                                    ; 22/02/2015 - dsectpm.s
2004                                    ; [ source: http://wiki.osdev.org/RTC ]
2005                                    ; read status register C to complete procedure
2006                                    ;(it is needed to get a next IRQ 8)
2007 00000B1D B00C                      mov    al, 0Ch ;
2008 00000B1F E670                      out    70h, al ; select register C
2009 00000B21 90                        nop
2010 00000B22 E471                      in     al, 71h ; just throw away contents
2011                                    ; 22/02/2015
2012 00000B24 B020                      MOV    AL,EOI       ; END OF INTERRUPT
2013                                    ;CLI                ; DISABLE INTERRUPTS TILL STACK CLEARED
2014 00000B26 E6A0                      OUT    INTB00,AL    ; FOR CONTROLLER #2
2015
2016                                    ; 23/05/2016
2017 00000B28 B020                      MOV    AL,EOI       ; GET END OF INTERRUPT MASK
2018 00000B2A FA                        CLI                 ; DISABLE INTERRUPTS TILL STACK CLEARED
2019 00000B2B E620                      OUT    INTA00,AL    ; END OF INTERRUPT TO 8259 - 1
2020                                    ;
2021                                    ; 23/05/2016
2022 00000B2D 20E4                      and    ah, ah
2023 00000B2F 0F84B7FEFFFF                jz      rtc_int_2
2024
2025                                    ; ah = 1 (half second)
2026 00000B35 58                        pop    eax ; *
2027 00000B36 1F                        pop    ds  ; **
2028 00000B37 CF                        iretd
2029
2030                                ; ////////////////
2031
2032                                    ; 28/08/2014
2033                                irq0:
2034 00000B38 6A00                        push       dword 0
2035 00000B3A EB48                      jmp    short which_irq
2036                                irq1:
2037 00000B3C 6A01                        push       dword 1
2038 00000B3E EB44                      jmp    short which_irq
2039                                irq2:
2040 00000B40 6A02                        push       dword 2
2041 00000B42 EB40                      jmp    short which_irq
2042                                irq3:
2043                                    ; 20/11/2015
2044                                    ; 24/10/2015
2045 00000B44 2EFF15[BD2F0100]          call   dword [cs:com2_irq3]
2046 00000B4B 6A03                      push   dword 3
2047 00000B4D EB35                      jmp    short which_irq
2048                                irq4:
2049                                    ; 20/11/2015
2050                                    ; 24/10/2015
2051 00000B4F 2EFF15[B92F0100]          call   dword [cs:com1_irq4]
2052 00000B56 6A04                        push       dword 4
2053 00000B58 EB2A                      jmp    short which_irq
2054                                irq5:
2055 00000B5A 6A05                        push       dword 5
2056 00000B5C EB26                      jmp    short which_irq
2057                                irq6:
2058 00000B5E 6A06                        push       dword 6
2059 00000B60 EB22                      jmp    short which_irq
2060                                irq7:
2061 00000B62 6A07                        push       dword 7
2062 00000B64 EB1E                      jmp    short which_irq
2063                                irq8:
2064 00000B66 6A08                        push       dword 8
2065 00000B68 EB1A                      jmp    short which_irq
2066                                irq9:
2067 00000B6A 6A09                        push       dword 9
2068 00000B6C EB16                      jmp    short which_irq
2069                                irq10:
2070 00000B6E 6A0A                        push       dword 10
2071 00000B70 EB12                      jmp    short which_irq
2072                                irq11:
```

```
2073 00000B72 6A0B                      push       dword 11
2074 00000B74 EB0E                    jmp    short which_irq
2075                                irq12:
2076 00000B76 6A0C                      push       dword 12
2077 00000B78 EB0A                    jmp    short which_irq
2078                                irq13:
2079 00000B7A 6A0D                      push       dword 13
2080 00000B7C EB06                    jmp    short which_irq
2081                                irq14:
2082 00000B7E 6A0E                      push       dword 14
2083 00000B80 EB02                    jmp    short which_irq
2084                                irq15:
2085 00000B82 6A0F                      push       dword 15
2086                                  ;jmp   short which_irq
2087
2088                                  ; 22/01/2017
2089                                  ; 19/10/2015
2090                                  ; 29/08/2014
2091                                  ; 21/08/2014
2092                                which_irq:
2093 00000B84 870424                  xchg   eax, [esp]  ; 28/08/2014
2094 00000B87 53                      push   ebx
2095 00000B88 56                      push   esi
2096 00000B89 57                      push   edi
2097 00000B8A 1E                      push   ds
2098 00000B8B 06                      push   es
2099                                  ;
2100 00000B8C 88C3                    mov    bl, al
2101                                  ;
2102 00000B8E B810000000              mov    eax, KDATA
2103 00000B93 8ED8                    mov    ds, ax
2104 00000B95 8EC0                    mov    es, ax
2105                                  ; 19/10/2015
2106 00000B97 FC                      cld
2107                                    ; 27/08/2014
2108 00000B98 8105[1A4A0100]A000-       add     dword [scr_row], 0A0h
2108 00000BA0 0000
2109                                  ;
2110 00000BA2 B417                    mov    ah, 17h      ; blue (1) background,
2111                                                 ; light gray (7) forecolor
2112 00000BA4 8B3D[1A4A0100]            mov     edi, [scr_row]
2113 00000BAA B049                    mov    al, 'I'
2114 00000BAC 66AB                    stosw
2115 00000BAE B052                    mov    al, 'R'
2116 00000BB0 66AB                    stosw
2117 00000BB2 B051                    mov    al, 'Q'
2118 00000BB4 66AB                    stosw
2119 00000BB6 B020                    mov    al, ' '
2120 00000BB8 66AB                    stosw
2121 00000BBA 88D8                    mov    al, bl
2122 00000BBC 3C0A                    cmp    al, 10
2123 00000BBE 7208                    jb     short ii1
2124 00000BC0 B031                    mov    al, '1'
2125 00000BC2 66AB                    stosw
2126 00000BC4 88D8                    mov    al, bl
2127 00000BC6 2C0A                    sub    al, 10
2128                                ii1:
2129 00000BC8 0430                    add    al, '0'
2130 00000BCA 66AB                    stosw
2131 00000BCC B020                    mov    al, ' '
2132 00000BCE 66AB                    stosw
2133 00000BD0 B021                    mov    al, '!'
2134 00000BD2 66AB                    stosw
2135 00000BD4 B020                    mov    al, ' '
2136 00000BD6 66AB                    stosw
2137                                  ; 23/02/2015
2138 00000BD8 80FB07                  cmp    bl, 7 ; check for IRQ 8 to IRQ 15
2139 00000BDB 7604                    jna    ii2
2140                                  ; 22/01/2017
2141 00000BDD B020                    mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2142 00000BDF E6A0                    out    0A0h, al ; the 2nd 8259
2143                                ii2:
2144 00000BE1 B020                    mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2145 00000BE3 E620                    out    20h, al ; the 2nd 8259
2146 00000BE5 E9CD010000              jmp     iiret
2147                                  ;
2148                                  ; 22/08/2014
2149                                  ;mov   al, 20h ; END OF INTERRUPT COMMAND TO 8259
2150                                  ;out   20h, al      ; 8259 PORT
2151                                  ;
2152                                  ;pop   es
2153                                  ;pop   ds
2154                                  ;pop   edi
2155                                  ;pop   esi
2156                                  ;pop   ebx
2157                                  ;pop   eax
2158                                  ;iret
2159
2160                                  ; 02/04/2015
2161                                  ; 25/08/2014
2162                                exc0:
2163 00000BEA 6A00                      push       dword 0
2164 00000BEC E990000000                jmp     cpu_except
2165                                exc1:
2166 00000BF1 6A01                      push       dword 1
2167 00000BF3 E989000000                jmp     cpu_except
2168                                exc2:
2169 00000BF8 6A02                      push       dword 2
2170 00000BFA E982000000                jmp     cpu_except
2171                                exc3:
2172 00000BFF 6A03                      push       dword 3
2173 00000C01 EB7E                      jmp     cpu_except
2174                                exc4:
2175 00000C03 6A04                      push       dword 4
2176 00000C05 EB7A                      jmp     cpu_except
```

```
2177                                    exc5:
2178 00000C07 6A05                             push       dword 5
2179 00000C09 EB76                             jmp     cpu_except
2180                                    exc6:
2181 00000C0B 6A06                             push       dword 6
2182 00000C0D EB72                             jmp     cpu_except
2183                                    exc7:
2184 00000C0F 6A07                             push       dword 7
2185 00000C11 EB6E                             jmp     cpu_except
2186                                    exc8:
2187                                          ; [esp] = Error code
2188 00000C13 6A08                             push       dword 8
2189 00000C15 EB5C                             jmp     cpu_except_en
2190                                    exc9:
2191 00000C17 6A09                             push       dword 9
2192 00000C19 EB66                             jmp     cpu_except
2193                                    exc10:
2194                                          ; [esp] = Error code
2195 00000C1B 6A0A                             push       dword 10
2196 00000C1D EB54                             jmp     cpu_except_en
2197                                    exc11:
2198                                          ; [esp] = Error code
2199 00000C1F 6A0B                             push       dword 11
2200 00000C21 EB50                             jmp     cpu_except_en
2201                                    exc12:
2202                                          ; [esp] = Error code
2203 00000C23 6A0C                             push       dword 12
2204 00000C25 EB4C                             jmp     cpu_except_en
2205                                    exc13:
2206                                          ; [esp] = Error code
2207 00000C27 6A0D                             push       dword 13
2208 00000C29 EB48                             jmp     cpu_except_en
2209                                    exc14:
2210                                          ; [esp] = Error code
2211 00000C2B 6A0E                             push       dword 14
2212 00000C2D EB44                           jmp    short cpu_except_en
2213                                    exc15:
2214 00000C2F 6A0F                             push       dword 15
2215 00000C31 EB4E                             jmp     cpu_except
2216                                    exc16:
2217 00000C33 6A10                             push       dword 16
2218 00000C35 EB4A                             jmp     cpu_except
2219                                    exc17:
2220                                          ; [esp] = Error code
2221 00000C37 6A11                             push       dword 17
2222 00000C39 EB38                           jmp    short cpu_except_en
2223                                    exc18:
2224 00000C3B 6A12                             push       dword 18
2225 00000C3D EB42                           jmp    short cpu_except
2226                                    exc19:
2227 00000C3F 6A13                             push       dword 19
2228 00000C41 EB3E                           jmp    short cpu_except
2229                                    exc20:
2230 00000C43 6A14                             push       dword 20
2231 00000C45 EB3A                           jmp    short cpu_except
2232                                    exc21:
2233 00000C47 6A15                             push       dword 21
2234 00000C49 EB36                           jmp    short cpu_except
2235                                    exc22:
2236 00000C4B 6A16                             push       dword 22
2237 00000C4D EB32                           jmp    short cpu_except
2238                                    exc23:
2239 00000C4F 6A17                             push       dword 23
2240 00000C51 EB2E                           jmp    short cpu_except
2241                                    exc24:
2242 00000C53 6A18                             push       dword 24
2243 00000C55 EB2A                           jmp    short cpu_except
2244                                    exc25:
2245 00000C57 6A19                             push       dword 25
2246 00000C59 EB26                           jmp    short cpu_except
2247                                    exc26:
2248 00000C5B 6A1A                             push       dword 26
2249 00000C5D EB22                           jmp    short cpu_except
2250                                    exc27:
2251 00000C5F 6A1B                             push       dword 27
2252 00000C61 EB1E                           jmp    short cpu_except
2253                                    exc28:
2254 00000C63 6A1C                             push       dword 28
2255 00000C65 EB1A                           jmp    short cpu_except
2256                                    exc29:
2257 00000C67 6A1D                             push       dword 29
2258 00000C69 EB16                           jmp    short cpu_except
2259                                    exc30:
2260 00000C6B 6A1E                             push       dword 30
2261 00000C6D EB04                           jmp    short cpu_except_en
2262                                    exc31:
2263 00000C6F 6A1F                             push       dword 31
2264 00000C71 EB0E                             jmp     short cpu_except
2265
2266                                          ; 19/10/2015
2267                                          ; 19/09/2015
2268                                          ; 01/09/2015
2269                                          ; 28/08/2015
2270                                          ; 28/08/2014
2271                                    cpu_except_en:
2272 00000C73 87442404                       xchg   eax, [esp+4] ; Error code
2273 00000C77 36A3[78050300]                 mov    [ss:error_code], eax
2274 00000C7D 58                             pop    eax  ; Exception number
2275 00000C7E 870424                         xchg   eax, [esp]
2276                                                ; eax = eax before exception
2277                                                ; [esp] -> exception number
2278                                                ; [esp+4] -> EIP to return
2279                                          ; 22/01/2017
2280                                          ; 19/10/2015
2281                                          ; 19/09/2015
```

```
                                        ; 01/09/2015
                                        ; 28/08/2015
                                        ; 29/08/2014
                                        ; 28/08/2014
                                        ; 25/08/2014
                                        ; 21/08/2014
                                cpu_except:  ; CPU Exceptions
00000C81 FC                             cld
00000C82 870424                         xchg   eax, [esp]
                                               ; eax = Exception number
                                               ; [esp] = eax (before exception)
00000C85 53                             push   ebx
00000C86 56                             push   esi
00000C87 57                             push   edi
00000C88 1E                             push   ds
00000C89 06                             push   es
                                        ; 28/08/2015
00000C8A 66BB1000                       mov    bx, KDATA
00000C8E 8EDB                           mov    ds, bx
00000C90 8EC3                           mov    es, bx
00000C92 0F20DB                         mov    ebx, cr3
00000C95 53                             push   ebx ; (*) page directory
                                        ; 19/10/2015
00000C96 FC                             cld
                                        ; 25/03/2015
00000C97 8B1D[188A0100]                 mov    ebx, [k_page_dir]
00000C9D 0F22DB                         mov    cr3, ebx
                                        ; 28/08/2015
00000CA0 83F80E                         cmp    eax, 0Eh ; 14, PAGE FAULT
00000CA3 750F                           jne    short cpu_except_nfp
00000CA5 E809520000                     call   page_fault_handler
00000CAA 21C0                           and    eax, eax
00000CAC 0F8401010000                     jz   iiretp ; 01/09/2015
00000CB2 B00E                           mov    al, 0Eh ; 14
                                cpu_except_nfp:
                                        ; 23/08/2016
00000CB4 803D[DA6F0000]03               cmp    byte [CRT_MODE], 3
00000CBB 7409                           je     short cpu_except_mode_3
00000CBD 50                             push   eax
00000CBE B003                           mov    al, 3
00000CC0 E8AB0E0000                     call   _set_mode
00000CC5 58                             pop    eax
                                cpu_except_mode_3:
                                        ; 02/04/2015
00000CC6 BB[13090000]                   mov    ebx, hang
00000CCB 875C241C                       xchg   ebx, [esp+28]
                                               ; EIP (points to instruction which faults)
                                               ; New EIP (hang)
00000CCF 891D[7C050300]                 mov    [FaultOffset], ebx
00000CD5 C744242008000000               mov    dword [esp+32], KCODE ; kernel's code segment
00000CDD 814C242400020000               or     dword [esp+36], 200h ; enable interrupts (set IF)
                                        ;
00000CE5 88C4                           mov    ah, al
00000CE7 240F                           and    al, 0Fh
00000CE9 3C09                           cmp    al, 9
00000CEB 7602                           jna    short h1ok
00000CED 0407                           add    al, 'A'-':'
                                h1ok:
00000CEF C0EC04                         shr    ah, 4
00000CF2 80FC09                         cmp    ah, 9
00000CF5 7603                           jna    short h2ok
00000CF7 80C407                         add    ah, 'A'-':'
                                h2ok:
00000CFA 86E0                           xchg   ah, al
00000CFC 66053030                       add    ax, '00'
00000D00 66A3[744C0100]                 mov    [excnstr], ax
                                        ;
                                        ; 29/08/2014
00000D06 A1[7C050300]                   mov    eax, [FaultOffset]
00000D0B 51                             push   ecx
00000D0C 52                             push   edx
00000D0D 89E3                           mov    ebx, esp
                                        ; 28/08/2015
00000D0F B910000000                     mov    ecx, 16        ; divisor value to convert binary number
                                                             ; to hexadecimal string
                                        ;mov   ecx, 10          ; divisor to convert
                                                               ; binary number to decimal string
                                b2d1:
00000D14 31D2                           xor    edx, edx
00000D16 F7F1                           div    ecx
00000D18 6652                           push   dx
00000D1A 39C8                           cmp    eax, ecx
00000D1C 73F6                           jnb    short b2d1
00000D1E BF[7F4C0100]                   mov    edi, EIPstr ; EIP value
                                                           ; points to instruction which faults
                                        ; 28/08/2015
00000D23 89C2                           mov    edx, eax
                                b2d2:
                                        ;add   al, '0'
00000D25 8A82[27430000]                 mov    al, [edx+hexchrs]
00000D2B AA                             stosb              ; write hexadecimal digit to its place
00000D2C 39E3                           cmp    ebx, esp
00000D2E 7606                           jna    short b2d3
00000D30 6658                           pop    ax
00000D32 88C2                           mov    dl, al
00000D34 EBEF                           jmp    short b2d2
                                b2d3:
00000D36 B068                           mov    al, 'h' ; 28/08/2015
00000D38 AA                             stosb
00000D39 B020                           mov    al, 20h          ; space
00000D3B AA                             stosb
00000D3C 30C0                           xor    al, al     ; to do it an ASCIIZ string
00000D3E AA                             stosb
                                        ;
00000D3F 5A                             pop    edx
```

```
2387 00000D40 59                            pop    ecx
2388                                        ;
2389 00000D41 B44F                          mov    ah, 4Fh      ; red (4) background,
2390                                                ; white (F) forecolor
2391 00000D43 BE[644C0100]                  mov    esi, exc_msg ; message offset
2392                                        ;
2393                                        ; 20/01/2017 (!cpu exception!)
2394                                        ;
2395 00000D48 8105[1A4A0100]A000-             add    dword [scr_row], 0A0h
2395 00000D50 0000
2396 00000D52 8B3D[1A4A0100]                  mov    edi, [scr_row]
2397                                        ;
2398 00000D58 C605[5B030300]00              mov    byte [sysflg], 0  ; system mode
2399 00000D5F FB                              sti
2400                                        ;
2401 00000D60 E8EDFBFFFF                    call   printk
2402                                        ;
2403 00000D65 B410                          mov    ah, 10h
2404 00000D67 E881010000                    call   int16h ; getc
2405                                        ;
2406 00000D6C B003                          mov    al, 3
2407 00000D6E E8FD0D0000                    call   _set_mode
2408                                        ;
2409 00000D73 B801000000                    mov    eax, 1
2410 00000D78 E916CD0000                    jmp    sysexit ; terminate process !!!
2411
2412                                        ; 22/01/2017
2413                                        ; 18/04/2016
2414                                        ; 28/08/2015
2415                                        ; 23/02/2015
2416                                        ; 20/08/2014
2417                                  ignore_int:
2418 00000D7D 50                            push   eax
2419 00000D7E 53                            push   ebx ; 23/02/2015
2420 00000D7F 56                            push   esi
2421 00000D80 57                            push   edi
2422 00000D81 1E                            push   ds
2423 00000D82 06                            push   es
2424                                        ; 18/04/2016
2425 00000D83 66B81000                      mov    ax, KDATA
2426 00000D87 8ED8                          mov    ds, ax
2427 00000D89 8EC0                          mov    es, ax
2428                                        ; 28/08/2015
2429 00000D8B 0F20D8                        mov    eax, cr3
2430 00000D8E 50                            push   eax ; (*) page directory
2431                                        ;
2432 00000D8F B467                          mov    ah, 67h      ; brown (6) background,
2433                                                ; light gray (7) forecolor
2434 00000D91 BE[2C4B0100]                  mov    esi, int_msg ; message offset
2435                                  piemsg:
2436                                          ; 27/08/2014
2437 00000D96 8105[1A4A0100]A000-             add     dword [scr_row], 0A0h
2437 00000D9E 0000
2438 00000DA0 8B3D[1A4A0100]                  mov     edi, [scr_row]
2439                                          ;
2440 00000DA6 E8A7FBFFFF                    call   printk
2441                                        ;
2442                                        ; 23/02/2015
2443 00000DAB B020                          mov    al, 20h  ; END OF INTERRUPT COMMAND TO
2444 00000DAD E6A0                          out    0A0h, al ; the 2nd 8259
2445                                        ; 22/08/2014
2446 00000DAF B020                          mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
2447 00000DB1 E620                          out    20h, al      ; 8259 PORT
2448                                  iiretp:
2449                                        ; 22/01/2017
2450                                        ; 01/09/2015
2451                                        ; 28/08/2015
2452 00000DB3 58                            pop    eax ; (*) page directory
2453 00000DB4 0F22D8                        mov    cr3, eax
2454                                  iiret:
2455 00000DB7 07                            pop    es
2456 00000DB8 1F                            pop    ds
2457 00000DB9 5F                            pop    edi
2458 00000DBA 5E                            pop    esi
2459 00000DBB 5B                            pop    ebx ; 29/08/2014
2460 00000DBC 58                            pop    eax
2461 00000DBD CF                            iretd
2462
2463                                        ; 23/05/2016
2464                                        ; 22/08/2014
2465                                        ; IBM PC/AT BIOS source code ----- 10/06/85 (bios.asm)
2466                                        ; (INT 1Ah)
2467                                        ;; Linux (v0.12) source code (main.c) by Linus Torvalds (1991)
2468                                  time_of_day:
2469 00000DBE E8DF5E0000                    call   UPD_IPR                    ; WAIT TILL UPDATE NOT IN PROGRESS
2470 00000DC3 726F                            jc      short time_of_day_retn ; 23/05/2016
2471 00000DC5 B000                          mov    al, CMOS_SECONDS
2472 00000DC7 E8F15E0000                    call   CMOS_READ
2473 00000DCC A2[888A0100]                  mov    [time_seconds], al
2474 00000DD1 B002                          mov    al, CMOS_MINUTES
2475 00000DD3 E8E55E0000                    call   CMOS_READ
2476 00000DD8 A2[898A0100]                  mov    [time_minutes], al
2477 00000DDD B004                          mov    al, CMOS_HOURS
2478 00000DDF E8D95E0000                    call   CMOS_READ
2479 00000DE4 A2[8A8A0100]                    mov     [time_hours], al
2480 00000DE9 B006                          mov    al, CMOS_DAY_WEEK
2481 00000DEB E8CD5E0000                    call   CMOS_READ
2482 00000DF0 A2[8B8A0100]                  mov    [date_wday], al
2483 00000DF5 B007                          mov    al, CMOS_DAY_MONTH
2484 00000DF7 E8C15E0000                    call   CMOS_READ
2485 00000DFC A2[8C8A0100]                  mov    [date_day], al
2486 00000E01 B008                          mov    al, CMOS_MONTH
2487 00000E03 E8B55E0000                    call   CMOS_READ
2488 00000E08 A2[8D8A0100]                  mov    [date_month], al
2489 00000E0D B009                          mov    al, CMOS_YEAR
```

```
2490 00000E0F E8A95E0000                call   CMOS_READ
2491 00000E14 A2[8E8A0100]              mov    [date_year], al
2492 00000E19 B032                      mov    al, CMOS_CENTURY
2493 00000E1B E89D5E0000                call   CMOS_READ
2494 00000E20 A2[8F8A0100]              mov    [date_century], al
2495                                    ;
2496 00000E25 B000                      mov    al, CMOS_SECONDS
2497 00000E27 E8915E0000                call   CMOS_READ
2498 00000E2C 3A05[888A0100]            cmp    al, [time_seconds]
2499 00000E32 758A                      jne    short time_of_day
2500
2501                                time_of_day_retn:
2502 00000E34 C3                        retn
2503
2504                                    ; 15/01/2017
2505                                    ; 10/06/2016
2506                                    ; 07/06/2016
2507                                    ; 06/06/2016
2508                                    ; 23/05/2016
2509                                rtc_p:
2510 00000E35 B101                      mov    cl, 1 ; 15/01/2017
2511 00000E37 EB02                      jmp    short rtc_p0
2512                                u_timer:
2513                                    ; Timer Events with 18.2 Hz Timer Ticks
2514                                    ; (and also timer events with RTC seconds)
2515 00000E39 28C9                      sub    cl, cl ; mov cl, 0 ; 15/01/2017
2516                                rtc_p0:
2517                                    ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
2518                                    ; Major Modification:
2519                                    ; Check and Perform Timer Events (for RTC)
2520                                    ; 25/08/2014 - 07/09/2014
2521                                    ; Retro UNIX 386 v1:
2522                                    ; Print Real Time Clock content
2523
2524                                    ; 15/01/2017
2525 00000E3B 880D[A8960100]            mov    byte [priority], cl ; 0 or 1 (not 2)
2526 00000E41 8A2D[AB960100]            mov    ch, [timer_events]
2527 00000E47 20ED                      and    ch, ch
2528 00000E49 7420                      jz     short rtc_p3
2529
2530 00000E4B BE[60040300]              mov    esi, timer_set  ; beginning address of
2531                                                            ; timer events space
2532                                rtc_p1:
2533 00000E50 8B06                      mov    eax, [esi]
2534 00000E52 20C0                      and    al, al ; 0 = free, >0 = process no.
2535 00000E54 7416                      jz     short rtc_p4
2536                                    ;
2537 00000E56 C1C810                    ror    eax, 16
2538                                    ; ah = response value, al = interrupt type
2539                                    ; 15/01/2017
2540                                    ; cl = interrupt source
2541                                    ;       1 = RTC, 0 = PIT
2542 00000E59 38C8                      cmp    al, cl
2543 00000E5B 750A                      jne    short rtc_p2 ; not as requested or undefined !
2544 00000E5D 3C01                      cmp    al, 1 ; 1 ; RTC interrupt ?
2545 00000E5F 7410                      je     short rtc_p5 ; yes, check for response
2546                                    ; 06/06/2016 - 18.2 Hz Timer Ticks
2547 00000E61 836E080A                  sub    dword [esi+8], 10 ; 1 tick = 10
2548 00000E65 7613                      jna    short rtc_p6  ; continue for responding
2549                                rtc_p2:
2550                                    ; 15/01/2017 (cl -> ch)
2551                                    ; 07/06/2016
2552 00000E67 FECD                      dec    ch    ; remain count of timer events
2553 00000E69 7501                      jnz    short rtc_p4
2554                                rtc_p3:
2555 00000E6B C3                        retn
2556                                rtc_p4:
2557                                    ;cmp   esi, timer_set + 240 ; 15*16 (last event)
2558                                    ;jnb   short rtc_p3 ; end of timer event space
2559 00000E6C 83C610                    add    esi, 16 ; next timer event
2560 00000E6F EBDF                      jmp    short rtc_p1
2561                                rtc_p5:
2562                                    ; current timer count ; 06/06/2016 (182)
2563 00000E71 816E08B6000000            sub    dword [esi+8], 182 ; 1 second (10*18.2)
2564 00000E78 77ED                      ja     short rtc_p2  ; check for the next
2565                                rtc_p6:
2566                                    ; it is the time of response!
2567 00000E7A 8B5E04                    mov    ebx, [esi+4] ; set (count limit) value
2568 00000E7D 895E08                    mov    [esi+8], ebx ; reset count down value
2569                                                        ; to count limit
2570                                    ; 19/12/2016
2571                                    ; 10/12/2016 - timer callback modification
2572 00000E80 8B7E0C                    mov    edi, [esi+12] ; response (or callback) address
2573 00000E83 807E0100                  cmp    byte [esi+1], 0 ; >0 = callback
2574 00000E87 762A                      jna    short rtc_p8
2575
2576                                    ; timer callback !
2577 00000E89 0FB61E                    movzx  ebx, byte [esi] ; process number (>0)
2578 00000E8C 89D8                      mov    eax, ebx
2579 00000E8E C0E302                    shl    bl, 2 ; *4
2580 00000E91 89BB[0C010300]            mov    [ebx+p.tcb-4], edi ; user's callback service addr
2581 00000E97 3A05[B3030300]            cmp    al, [u.uno]
2582 00000E9D 7521                      jne    short rtc_p9
2583 00000E9F 893D[D0030300]            mov    [u.tcb], edi
2584                                rtc_p7:
2585                                    ; 15/01/2017
2586 00000EA5 B002                      mov    al, 2
2587 00000EA7 A2[A8960100]              mov    [priority], al ; 2
2588                                    ; 10/01/2017
2589                                    ;mov   byte [u.pri], 2
2590 00000EAC A2[A9030300]              mov    [u.pri], al ; 2
2591 00000EB1 EBB4                      jmp    short rtc_p2
2592                                rtc_p8:
2593                                    ; response address is physical address of
2594                                    ; the program's response (signal return) byte
```

```
                                        ; 06/06/2016
                                        ;mov   edi, [esi+12] ; response address
00000EB3 8827                           mov    [edi], ah     ; response value
                                        ;
00000EB5 C1C010                         rol    eax, 16
                                        ; 15/01/2017
00000EB8 3A05[B3030300]                 cmp    al, [u.uno] ; running process ?
00000EBE 74E5                           je     short rtc_p7
                                rtc_p9:
                                        ; al = process number  ; 10/06/2016
00000EC0 B202                           mov    dl, 2 ; priority, 2 = event (high)
00000EC2 E8EC1D0100                     call   set_run_sequence ; 19/05/2016
00000EC7 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:
00000EC9 6650                           push   ax
00000ECB B00B                           mov    al, 0Bh  ; In-Service register
00000ECD E620                           out    20h, al
00000ECF EB00                             jmp short $+2
00000ED1 EB00                           jmp short $+2
00000ED3 E420                           in     al, 20h
00000ED5 2480                           and    al, 80h ; bit 7 (is it real IRQ 7 or fake?)
00000ED7 7404                             jz      short irq7_iret ; Fake (spurious) IRQ, do not send EOI
00000ED9 B020                             mov     al, 20h ; EOI
00000EDB E620                           out    20h, al
                                irq7_iret:
00000EDD 6658                           pop    ax
00000EDF 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)
                                        ;
00000EE0 D410                           db 0D4h,10h                     ; Undocumented inst. AAM
                                                                ; AH = AL / 10h
                                                                ; AL = AL MOD 10h
00000EE2 660D3030                       or ax,'00'                      ; Make it ASCII based

00000EE6 86E0                             xchg ah, al

00000EE8 C3                             retn

                                ; 15/12/2020
                                real_mem_16m_64k:
00000EE9 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:
00000EEB 0000                           dw 0   ; HW of default LFB addr (for mode 118h)


                                    %include 'keyboard.s' ; 07/03/2015
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - keyboard.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 15/01/2017
                                <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> ; 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:
00000EED 9C                     <1>       pushfd ; 28/08/2014
```

```
00000EEE 0E                <1>       push   cs
00000EEF E801000000        <1>       call   KEYBOARD_IO_1 ; getc_int
00000EF4 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               :
                           <1>       ;                  02H        24.0        12H        6.0               :
                           <1>       ;                  03H        21.8        13H        5.5               :
                           <1>       ;                  04H        20.0        14H        5.0               :
                           <1>       ;                  05H        18.5        15H        4.6               :
                           <1>       ;                  06H        17.1        16H        4.3               :
                           <1>       ;                  07H        16.0        17H        4.0               :
                           <1>       ;                  08H        15.0        18H        3.7               :
                           <1>       ;                  09H        13.3        19H        3.3               :
                           <1>       ;                  0AH        12.0        1AH        3.0               :
                           <1>       ;                  0BH        10.9        1BH        2.7               :
                           <1>       ;                  0CH        10.0        1CH        2.5               :
                           <1>       ;                  0DH         9.2        1DH        2.3               :
                           <1>       ;                  0EH         8.6        1EH        2.1               :
                           <1>       ;                  0FH         8.0        1FH        2.0               :
                           <1>       ;                                                                       :
                           <1>       ;           (BH) = TYPAMATIC DELAY  (BITS 2 - 7 MUST BE RESET TO 0)       :
                           <1>       ;                                                                      :
                           <1>       ;                  REGISTER     DELAY                                    :
                           <1>       ;                   VALUE       VALUE                                    :
                           <1>       ;                  ------------------                                   :
                           <1>       ;                  00H        250 ms                                    :
                           <1>       ;                  01H        500 ms                                    :
                           <1>       ;                  02H        750 ms                                    :
                           <1>       ;                  03H       1000 ms                                    :
                           <1>       ;--------------------------------------------------------------------------:
                           <1>       ;     (AH)= 05H  PLACE ASCII CHARACTER/SCAN CODE COMBINATION IN KEYBOARD    :
                           <1>       ;               BUFFER AS IF STRUCK FROM KEYBOARD                          :
                           <1>       ;               ENTRY:  (CL) = ASCII CHARACTER                            :
                           <1>       ;                       (CH) = SCAN CODE                                   :
                           <1>       ;               EXIT:   (AH) = 00H = SUCCESSFUL OPERATION                  :
                           <1>       ;                       (AL) = 01H = UNSUCCESSFUL - BUFFER FULL            :
                           <1>       ;               FLAGS:  CARRY IF ERROR                                     :
                           <1>       ;--------------------------------------------------------------------------:
                           <1>       ;     (AH)= 10H  EXTENDED READ INTERFACE FOR THE ENHANCED KEYBOARD,         :
                           <1>       ;               OTHERWISE SAME AS FUNCTION AH=0                            :
                           <1>       ;--------------------------------------------------------------------------:
                           <1>       ;     (AH)= 11H  EXTENDED ASCII STATUS FOR THE ENHANCED KEYBOARD,          :
                           <1>       ;               OTHERWISE SAME AS FUNCTION AH=1                            :
                           <1>       ;--------------------------------------------------------------------------:
                           <1>       ;     (AH)= 12H  RETURN THE EXTENDED SHIFT STATUS IN AX REGISTER           :
                           <1>       ;               AL = BITS FROM KB_FLAG, AH = BITS FOR LEFT AND RIGHT       :
                           <1>       ;               CTL AND ALT KEYS FROM KB_FLAG_1 AND KB_FLAG_3              :
                           <1>       ; OUTPUT                                                                :
                           <1>       ;     AS NOTED ABOVE, ONLY (AX) AND FLAGS CHANGED                        :
                           <1>       ;     ALL REGISTERS RETAINED                                              :
                           <1>       ;--------------------------------------------------------------------------
                           <1>
                           <1> ; 15/01/2017
                           <1> ; 14/01/2017
                           <1> ; 02/01/2017
                           <1> ; 29/05/2016
                           <1> ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                           <1> int32h:  ; Keyboard BIOS
                           <1>
                           <1> KEYBOARD_IO_1:
                           <1>       ;sti                    ; INTERRUPTS BACK ON
                           <1>       ; 29/05/2016
```

```
00000EF5 80642408BE          <1>         and     byte [esp+8], 10111110b ; clear zero flag and cary flag
                             <1>       ;
00000EFA 1E                  <1>       push   ds                 ; SAVE CURRENT DS
00000EFB 53                  <1>       push   ebx                ; SAVE BX TEMPORARILY
                             <1>       ;push  ecx                ; SAVE CX TEMPORARILY
00000EFC 66BB1000            <1>         mov     bx, KDATA
00000F00 8EDB                <1>       mov    ds, bx             ; PUT SEGMENT VALUE OF DATA AREA INTO DS
                             <1>
                             <1>       ; 14/01/2017
00000F02 8B1C24              <1>       mov    ebx, [esp]
                             <1>       ;; 15/01/2017
                             <1>       ; 02/01/2017
                             <1>       ;;mov  byte [intflg], 32h  ; keyboard interrupt
00000F05 FB                  <1>       sti
                             <1>       ;
                             <1>
00000F06 08E4                <1>       or     ah, ah             ; CHECK FOR (AH)= 00H
00000F08 743A                <1>       jz     short _K1          ; ASCII_READ
00000F0A FECC                <1>       dec    ah                    ; CHECK FOR (AH)= 01H
00000F0C 7453                <1>         jz      short _K2             ; ASCII_STATUS
00000F0E FECC                <1>       dec    ah                 ; CHECK FOR (AH)= 02H
00000F10 0F8494000000        <1>         jz      _K3                   ; SHIFT STATUS
00000F16 FECC                <1>       dec    ah                 ; CHECK FOR (AH)= 03H
00000F18 0F8493000000        <1>         jz      _K300                 ; SET TYPAMATIC RATE/DELAY
00000F1E 80EC02              <1>       sub    ah, 2              ; CHECK FOR (AH)= 05H
00000F21 0F84BC000000        <1>         jz      _K500                 ; KEYBOARD WRITE
                             <1> _KIO1:
00000F27 80EC0B              <1>       sub    ah, 11             ; AH =  10H
00000F2A 740C                <1>       jz     short _K1E         ; EXTENDED ASCII READ
00000F2C FECC                <1>       dec    ah                 ; CHECK FOR (AH)= 11H
00000F2E 7422                <1>       jz     short _K2E         ; EXTENDED_ASCII_STATUS
00000F30 FECC                <1>       dec    ah                 ; CHECK FOR (AH)= 12H
00000F32 7458                <1>       jz     short _K3E         ; EXTENDED_SHIFT_STATUS
                             <1> _KIO_EXIT:
                             <1>       ; 02/01/2017
00000F34 FA                  <1>       cli
                             <1>       ;;mov  byte [intflg], 0 ;; 15/01/2017
                             <1>       ;
                             <1>       ;pop   ecx                ; RECOVER REGISTER
00000F35 5B                  <1>       pop    ebx                ; RECOVER REGISTER
00000F36 1F                  <1>       pop    ds                 ; RECOVER SEGMENT
00000F37 CF                  <1>       iretd                     ; INVALID COMMAND, EXIT
                             <1>
                             <1>       ;----- ASCII CHARACTER
                             <1> _K1E:
00000F38 E8D3000000          <1>       call   _K1S               ; GET A CHARACTER FROM THE BUFFER (EXTENDED)
00000F3D E848010000          <1>       call   _KIO_E_XLAT        ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000F42 EBF0                <1>       jmp    short _KIO_EXIT        ; GIVE IT TO THE CALLER
                             <1> _K1:
00000F44 E8C7000000          <1>       call   _K1S               ; GET A CHARACTER FROM THE BUFFER
00000F49 E847010000          <1>       call   _KIO_S_XLAT        ; ROUTINE TO XLATE FOR STANDARD CALLS
00000F4E 72F4                <1>       jc     short _K1          ; CARRY SET MEANS TROW CODE AWAY
                             <1> _K1A:
00000F50 EBE2                <1>       jmp    short _KIO_EXIT         ; RETURN TO CALLER
                             <1>
                             <1>       ;----- ASCII STATUS
                             <1> _K2E:
00000F52 E804010000          <1>       call   _K2S               ; TEST FOR CHARACTER IN BUFFER (EXTENDED)
00000F57 7420                <1>       jz     short _K2B         ; RETURN IF BUFFER EMPTY
00000F59 9C                  <1>       pushf                     ; SAVE ZF FROM TEST
00000F5A E82B010000          <1>       call   _KIO_E_XLAT        ; ROUTINE TO XLATE FOR EXTENDED CALLS
00000F5F EB17                <1>       jmp    short _K2A          ; GIVE IT TO THE CALLER
                             <1> _K2:
00000F61 E8F5000000          <1>       call   _K2S               ; TEST FOR CHARACTER IN BUFFER
00000F66 7411                <1>       jz     short _K2B         ; RETURN IF BUFFER EMPTY
00000F68 9C                  <1>       pushf                     ; SAVE ZF FROM TEST
00000F69 E827010000          <1>       call   _KIO_S_XLAT        ; ROUTINE TO XLATE FOR STANDARD CALLS
00000F6E 7308                <1>       jnc    short _K2A          ; CARRY CLEAR MEANS PASS VALID CODE
00000F70 9D                  <1>       popf                      ; INVALID CODE FOR THIS TYPE OF CALL
00000F71 E89A000000          <1>       call   _K1S               ; THROW THE CHARACTER AWAY
00000F76 EBE9                <1>       jmp    short _K2          ; GO LOOK FOR NEXT CHAR, IF ANY
                             <1>  K2A:
00000F78 9D                  <1>       popf                      ; RESTORE ZF FROM TEST
                             <1> _K2B:
                             <1>       ; 02/01/2017
00000F79 FA                  <1>       cli
                             <1>       ;; mov byte [intflg], 0 ;; 15/01/2017
                             <1>       ;
                             <1>       ;pop   ecx                ; RECOVER REGISTER
00000F7A 5B                  <1>       pop    ebx                ; RECOVER REGISTER
00000F7B 1F                  <1>       pop    ds                 ; RECOVER SEGMENT
                             <1>       ; (*) 29/05/2016
                             <1>       ; (*) retf 4              ; THROW AWAY (e)FLAGS
00000F7C 7208                <1>       jc     short _k2d
00000F7E 7505                <1>       jnz    short _k2c
00000F80 804C240840          <1>       or     byte [esp+8], 01000000b   ; set zero flag bit of eflags register
                             <1> _k2c:
00000F85 CF                  <1>       iretd
                             <1> _k2d:
                             <1>       ; 29/05/2016 -set carry flag on stack-
                             <1>       ; [esp] = EIP
                             <1>       ; [esp+4] = CS
                             <1>       ; [esp+8] = E-FLAGS
00000F86 804C240801          <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
                             <1>       ; [esp+12] = ESP (user)
                             <1>       ; [esp+16] = SS (User)
00000F8B CF                  <1>       iretd
                             <1>
                             <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>       ;----- SHIFT STATUS
                        <1> _K3E:                                   ; GET THE EXTENDED SHIFT STATUS FLAGS
00000F8C 8A25[A66F0000] <1>       mov   ah, [KB_FLAG_1]            ; GET SYSTEM SHIFT KEY STATUS
00000F92 80E404         <1>       and   ah, SYS_SHIFT       ; MASK ALL BUT SYS KEY BIT
                        <1>       ;mov  cl, 5               ; SHIFT THEW SYSTEMKEY BIT OVER TO
                        <1>       ;shl  ah, cl              ; BIT 7 POSITION
00000F95 C0E405         <1>         shl ah, 5
00000F98 A0[A66F0000]   <1>       mov   al, [KB_FLAG_1]            ; GET SYSTEM SHIFT STATES BACK
00000F9D 2473           <1>       and   al, 01110011b       ; ELIMINATE SYS SHIFT, HOLD_STATE AND INS_SHIFT
00000F9F 08C4           <1>       or    ah, al                 ; MERGE REMAINING BITS INTO AH
00000FA1 A0[A86F0000]   <1>       mov   al, [KB_FLAG_3]            ; GET RIGHT CTL AND ALT
00000FA6 240C           <1>       and   al, 00001100b       ; ELIMINATE LC_E0 AND LC_E1
00000FA8 08C4           <1>       or    ah, al              ; OR THE SHIFT FLAGS TOGETHER
                        <1> _K3:
00000FAA A0[A56F0000]   <1>       mov   al, [KB_FLAG]       ; GET THE SHIFT STATUS FLAGS
                        <1>       ;jmp  short _KIO_EXIT           ; RETURN TO CALLER
00000FAF EB83           <1>       jmp   _KIO_EXIT
                        <1>
                        <1>       ;----- SET TYPAMATIC RATE AND DELAY
                        <1>  K300:
00000FB1 3C05           <1>       cmp   al, 5               ; CORRECT FUNCTION CALL?
                        <1>       ;jne  short _KIO_EXIT           ; NO, RETURN
00000FB3 0F857BFFFFFF   <1>       jne   _KIO_EXIT
00000FB9 F6C3E0         <1>       test  bl, 0E0h            ; TEST FOR OUT-OF-RANGE RATE
00000FBC 0F8572FFFFFF   <1>         jnz   _KIO_EXIT                 ; RETURN IF SO
00000FC2 F6C7FC         <1>       test  BH, 0FCh            ; TEST FOR OUT-OF-RANGE DELAY
00000FC5 0F8569FFFFFF   <1>         jnz   _KIO_EXIT                 ; RETURN IF SO
00000FCB B0F3           <1>       mov   al, KB_TYPA_RD            ; COMMAND FOR TYPAMATIC RATE/DELAY
00000FCD E8DA060000     <1>       call  SND_DATA            ; SEND TO KEYBOARD
                        <1>       ;mov  cx, 5               ; SHIFT COUNT
                        <1>       ;shl  bh, cl              ; SHIFT DELAY OVER
00000FD2 C0E705         <1>       shl   bh, 5
00000FD5 88D8           <1>       mov   al, bl              ; PUT IN RATE
00000FD7 08F8           <1>       or    al, bh              ; AND DELAY
00000FD9 E8CE060000     <1>       call  SND_DATA            ; SEND TO KEYBOARD
00000FDE E951FFFFFF     <1>         jmp   _KIO_EXIT                 ; RETURN TO CALLER
                        <1>
                        <1>       ;----- WRITE TO KEYBOARD BUFFER
                        <1> _K500:
00000FE3 56             <1>       push  esi                 ; SAVE SI (esi)
00000FE4 FA             <1>       cli                       ;
00000FE5 8B1D[B66F0000] <1>       mov   ebx, [BUFFER_TAIL]  ; GET THE 'IN TO' POINTER TO THE BUFFER
00000FEB 89DE           <1>       mov   esi, ebx            ; SAVE A COPY IN CASE BUFFER NOT FULL
00000FED E8D3000000     <1>       call  _K4                 ; BUMP THE POINTER TO SEE IF BUFFER IS FULL
00000FF2 3B1D[B26F0000] <1>       cmp   ebx, [BUFFER_HEAD]  ; WILL THE BUFFER OVERRUN IF WE STORE THIS?
00000FF8 740D           <1>       je    short _K502         ; YES - INFORM CALLER OF ERROR
00000FFA 66890E         <1>       mov   [esi], cx           ; NO - PUT ASCII/SCAN CODE INTO BUFFER
00000FFD 891D[B66F0000] <1>       mov   [BUFFER_TAIL], ebx  ; ADJUST 'IN TO' POINTER TO REFLECT CHANGE
00001003 28C0           <1>       sub   al, al              ; TELL CALLER THAT OPERATION WAS SUCCESSFUL
00001005 EB02           <1>       jmp   short _K504         ; SUB INSTRUCTION ALSO RESETS CARRY FLAG
                        <1> _K502:
00001007 B001           <1>       mov   al, 01h                   ; BUFFER FULL INDICATION
                        <1> _K504:
00001009 FB             <1>       sti
0000100A 5E             <1>       pop   esi                 ; RECOVER SI (esi)
0000100B 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:
00001010 FA             <1>       cli   ; 03/12/2014
00001011 8B1D[B26F0000] <1>         mov   ebx, [BUFFER_HEAD]    ; GET POINTER TO HEAD OF BUFFER
00001017 3B1D[B66F0000] <1>         cmp   ebx, [BUFFER_TAIL]    ; TEST END OF BUFFER
                        <1>       ;jne  short _K1U          ; IF ANYTHING IN BUFFER SKIP INTERRUPT
0000101D 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
0000101F FB             <1>       sti                       ; INTERRUPTS BACK ON DURING LOOP
00001020 90             <1>       nop                       ; ALLOW AN INTERRUPT TO OCCUR
                        <1> _K1U:
00001021 FA             <1>       cli                       ; INTERRUPTS BACK OFF
00001022 8B1D[B26F0000] <1>         mov       ebx, [BUFFER_HEAD]  ; GET POINTER TO HEAD OF BUFFER
00001028 3B1D[B66F0000] <1>         cmp     ebx, [BUFFER_TAIL]    ; TEST END OF BUFFER
                        <1> _k1x:
0000102E 53             <1>       push  ebx                 ; SAVE ADDRESS
0000102F 9C             <1>       pushf                     ; SAVE FLAGS
00001030 E82F070000     <1>       call  MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
00001035 8A1D[A76F0000] <1>       mov   bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
0000103B 30C3           <1>       xor   bl, al              ; SEE IF ANY DIFFERENT
0000103D 80E307         <1>       and   bl, 07h      ; KB_LEDS    ; ISOLATE INDICATOR BITS
00001040 7406           <1>       jz    short _K1V          ; IF NO CHANGE BYPASS UPDATE
00001042 E8C9060000     <1>       call  SND_LED1
00001047 FA             <1>       cli                       ; DISABLE INTERRUPTS
                        <1> _K1V:
00001048 9D             <1>       popf                      ; RESTORE FLAGS
00001049 5B             <1>       pop   ebx                 ; RESTORE ADDRESS
0000104A 74D3           <1>         je      short _K1T              ; LOOP UNTIL SOMETHING IN BUFFER
                        <1>       ;
```

```
0000104C 668B03              <1>       mov    ax, [ebx]           ; GET SCAN CODE AND ASCII CODE
0000104F E871000000          <1>         call    _K4                    ; MOVE POINTER TO NEXT POSITION
00001054 891D[B26F0000]      <1>         mov     [BUFFER_HEAD], ebx     ; STORE VALUE IN VARIABLE
0000105A C3                  <1>       retn                        ; RETURN
                             <1>
                             <1>       ;----- READ THE KEY TO SEE IF ONE IS PRESENT -----
                             <1> _K2S:
0000105B FA                  <1>       cli                         ; INTERRUPTS OFF
0000105C 8B1D[B26F0000]      <1>         mov     ebx, [BUFFER_HEAD]     ; GET HEAD POINTER
00001062 3B1D[B66F0000]      <1>         cmp     ebx, [BUFFER_TAIL]     ; IF EQUAL (Z=1) THEN NOTHING THERE
00001068 668B03              <1>       mov    ax, [ebx]
0000106B 9C                  <1>       pushf                       ; SAVE FLAGS
0000106C 6650                <1>       push   ax                   ; SAVE CODE
0000106E E8F1060000          <1>       call   MAKE_LED             ; GO GET MODE INDICATOR DATA BYTE
00001073 8A1D[A76F0000]      <1>       mov    bl, [KB_FLAG_2]      ; GET PREVIOUS BITS
00001079 30C3                <1>       xor    bl, al               ; SEE IF ANY DIFFERENT
0000107B 80E307              <1>       and    bl, 07h ; KB_LEDS    ; ISOLATE INDICATOR BITS
0000107E 7405                <1>       jz     short _K2T           ; IF NO CHANGE BYPASS UPDATE
00001080 E874060000          <1>       call   SND_LED                    ; GO TURN ON MODE INDICATORS
                             <1> _K2T:
00001085 6658                <1>       pop    ax                   ; RESTORE CODE
00001087 9D                  <1>       popf                        ; RESTORE FLAGS
00001088 FB                  <1>       sti                         ; INTERRUPTS BACK ON
00001089 C3                  <1>       retn                        ; RETURN
                             <1>
                             <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR EXTENDED CALLS -----
                             <1> _KIO_E_XLAT:
0000108A 3CF0                <1>       cmp    al, 0F0h             ; IS IT ONE OF THE FILL-INs?
0000108C 7506                <1>       jne    short _KIO_E_RET     ; NO, PASS IT ON
0000108E 08E4                <1>         or   ah, ah               ; AH = 0 IS SPECIAL CASE
00001090 7402                <1>         jz   short _KIO_E_RET        ; PASS THIS ON UNCHANGED
00001092 30C0                <1>       xor    al, al               ; OTHERWISE SET AL = 0
                             <1> _KIO_E_RET:
00001094 C3                  <1>       retn                        ; GO BACK
                             <1>
                             <1>       ;----- ROUTINE TO TRANSLATE SCAN CODE PAIRS FOR STANDARD CALLS -----
                             <1> _KIO_S_XLAT:
00001095 80FCE0              <1>       cmp    ah, 0E0h             ; IS IT KEYPAD ENTER OR / ?
00001098 750F                <1>       jne    short _KIO_S2        ; NO, CONTINUE
0000109A 3C0D                <1>       cmp    al, 0Dh                    ; KEYPAD ENTER CODE?
0000109C 7408                <1>         je   short _KIO_S1        ; YES, MASSAGE A BIT
0000109E 3C0A                <1>       cmp    al, 0Ah                    ; CTRL KEYPAD ENTER CODE?
000010A0 7404                <1>         je   short _KIO_S1        ; YES, MASSAGE THE SAME
000010A2 B435                <1>       mov    ah, 35h                    ; NO, MUST BE KEYPAD /
                             <1> _kio_ret: ; 03/12/2014
000010A4 F8                  <1>       clc
000010A5 C3                  <1>       retn
                             <1>       ;jmp   short _KIO_USE             ; GIVE TO CALLER
                             <1> _KIO_S1:
000010A6 B41C                <1>       mov    ah, 1Ch                    ; CONVERT TO COMPATIBLE OUTPUT
                             <1>       ;jmp   short _KIO_USE             ; GIVE TO CALLER
000010A8 C3                  <1>       retn
                             <1> _KIO_S2:
000010A9 80FC84              <1>       cmp    ah, 84h                    ; IS IT ONE OF EXTENDED ONES?
000010AC 7715                <1>       ja     short _KIO_DIS             ; YES, THROW AWAY AND GET ANOTHER CHAR
000010AE 3CF0                <1>       cmp    al, 0F0h             ; IS IT ONE OF THE FILL-INs?
000010B0 7506                <1>         jne  short _KIO_S3        ; NO, TRY LAST TEST
000010B2 08E4                <1>       or     ah, ah               ; AH = 0 IS SPECIAL CASE
000010B4 740C                <1>         jz   short _KIO_USE             ; PASS THIS ON UNCHANGED
000010B6 EB0B                <1>       jmp    short _KIO_DIS             ; THROW AWAY THE REST
                             <1> _KIO_S3:
000010B8 3CE0                <1>       cmp    al, 0E0h             ; IS IT AN EXTENSION OF A PREVIOUS ONE?
                             <1>       ;jne   short _KIO_USE             ; NO, MUST BE A STANDARD CODE
000010BA 75E8                <1>       jne    short _kio_ret
000010BC 08E4                <1>       or     ah, ah               ; AH = 0 IS SPECIAL CASE
000010BE 7402                <1>         jz   short _KIO_USE             ; JUMP IF AH = 0
000010C0 30C0                <1>       xor    al, al               ; CONVERT TO COMPATIBLE OUTPUT
                             <1>       ;jmp   short _KIO_USE             ; PASS IT ON TO CALLER
                             <1> _KIO_USE:
                             <1>       ;clc                        ; CLEAR CARRY TO INDICATE GOOD CODE
000010C2 C3                  <1>       retn                        ; RETURN
                             <1> _KIO_DIS:
000010C3 F9                  <1>       stc                         ; SET CARRY TO INDICATE DISCARD CODE
000010C4 C3                  <1>       retn                        ; RETURN
                             <1>
                             <1>       ;----- INCREMENT BUFFER POINTER ROUTINE -----
                             <1> _K4:
000010C5 43                  <1>       inc     ebx
000010C6 43                  <1>       inc    ebx                  ; MOVE TO NEXT WORD IN LIST
000010C7 3B1D[AE6F0000]      <1>         cmp     ebx, [BUFFER_END]      ; AT END OF BUFFER?
                             <1>         ;jne    short _K5               ; NO, CONTINUE
000010CD 7206                <1>       jb     short _K5
000010CF 8B1D[AA6F0000]      <1>         mov     ebx, [BUFFER_START]     ; YES, RESET TO BUFFER BEGINNING
                             <1> _K5:
000010D5 C3                  <1>       retn
                             <1>
                             <1> ; 20/02/2015
                             <1> ; 05/12/2014
                             <1> ; 26/08/2014
                             <1> ; KEYBOARD (HARDWARE) INTERRUPT -  IRQ LEVEL 1
                             <1> ; (INT_09h - Retro UNIX 8086 v1 - U9.ASM, 07/03/2014)
                             <1> ;
                             <1> ; Derived from "KB_INT_1" procedure of IBM "pc-at"
                             <1> ; rombios source code (06/10/1985)
                             <1> ; 'keybd.asm', HARDWARE INT 09h - (IRQ Level 1)
                             <1>
                             <1> ; EQUATES (IBM PC-XT-286 BIOS, 1986, 'POSQEQU.INC')
                             <1>
                             <1> ;--------- 8042 COMMANDS --------------------------------------------------------
                             <1> ENA_KBD            equ    0AEh         ; ENABLE KEYBOARD COMMAND
                             <1> DIS_KBD            equ    0ADh         ; DISABLE KEYBOARD COMMAND
                             <1> SHUT_CMD    equ    0FEh         ; CAUSE A SHUTDOWN COMMAND
                             <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
                             <1> STATUS_PORT equ    064h         ; 8042 STATUS PORT
                             <1> INPT_BUF_FULL      equ    00000010b    ; 1 = +INPUT BUFFER FULL
```

```
                        <1> PORT_A              equ    060h          ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
                        <1> ;---------- 8042 KEYBOARD RESPONSE ----------------------------------------------
                        <1> KB_ACK              equ    0FAh          ; ACKNOWLEDGE PROM TRANSMISSION
                        <1> KB_RESEND    equ    0FEh          ; RESEND REQUEST
                        <1> KB_OVER_RUN  equ    0FFh          ; OVER RUN SCAN CODE
                        <1> ;---------- KEYBOARD/LED COMMANDS -------------------------------------------------
                        <1> KB_ENABLE    equ    0F4h          ; KEYBOARD ENABLE
                        <1> LED_CMD             equ    0EDh          ; LED WRITE COMMAND
                        <1> KB_TYPA_RD   equ    0F3h          ; TYPAMATIC RATE/DELAY COMMAND
                        <1> ;---------- KEYBOARD SCAN CODES ---------------------------------------------------
                        <1> NUM_KEY             equ    69            ; SCAN CODE FOR      NUMBER LOCK KEY
                        <1> SCROLL_KEY   equ    70            ; SCAN CODE FOR      SCROLL LOCK KEY
                        <1> ALT_KEY             equ    56            ; SCAN CODE FOR      ALTERNATE SHIFT KEY
                        <1> CTL_KEY             equ    29            ; SCAN CODE FOR      CONTROL KEY
                        <1> CAPS_KEY     equ    58            ; SCAN CODE FOR      SHIFT LOCK KEY
                        <1> DEL_KEY             equ    83            ; SCAN CODE FOR      DELETE KEY
                        <1> INS_KEY             equ    82            ; SCAN CODE FOR      INSERT KEY
                        <1> LEFT_KEY     equ    42            ; SCAN CODE FOR      LEFT SHIFT
                        <1> RIGHT_KEY    equ    54            ; SCAN CODE FOR      RIGHT SHIFT
                        <1> SYS_KEY             equ    84            ; SCAN CODE FOR      SYSTEM KEY
                        <1> ;---------- ENHANCED KEYBOARD SCAN CODES ------------------------------------------
                        <1> ID_1         equ    0ABh          ; 1ST ID CHARACTER FOR KBX
                        <1> ID_2         equ    041h          ; 2ND ID CHARACTER FOR KBX
                        <1> ID_2A        equ    054h          ; ALTERNATE 2ND ID CHARACTER FOR KBX
                        <1> F11_M        equ    87            ; F11 KEY MAKE
                        <1> F12_M        equ    88            ; F12 KEY MAKE
                        <1> MC_E0        equ    224           ; GENERAL MARKER CODE
                        <1> MC_E1        equ    225           ; PAUSE KEY MARKER CODE
                        <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG-------------------------------------------
                        <1> RIGHT_SHIFT  equ    00000001b     ; RIGHT SHIFT KEY DEPRESSED
                        <1> LEFT_SHIFT   equ    00000010b     ; LEFT SHIFT KEY DEPRESSED
                        <1> CTL_SHIFT    equ    00000100b     ; CONTROL SHIFT KEY DEPRESSED
                        <1> ALT_SHIFT    equ    00001000b     ; ALTERNATE SHIFT KEY DEPRESSED
                        <1> SCROLL_STATE equ    00010000b     ; SCROLL LOCK STATE IS ACTIVE
                        <1> NUM_STATE    equ    00100000b     ; NUM LOCK STATE IS ACTIVE
                        <1> CAPS_STATE   equ    01000000b     ; CAPS LOCK STATE IS ACTIVE
                        <1> INS_STATE    equ    10000000b     ; INSERT STATE IS ACTIVE
                        <1> ;---------- FLAG EQUATES WITHIN @KB_FLAG_1 -----------------------------------------
                        <1> L_CTL_SHIFT  equ    00000001b     ; LEFT CTL KEY DOWN
                        <1> L_ALT_SHIFT  equ    00000010b     ; LEFT ALT KEY DOWN
                        <1> SYS_SHIFT    equ    00000100b     ; SYSTEM KEY DEPRESSED AND HELD
                        <1> HOLD_STATE   equ    00001000b     ; SUSPEND KEY HAS BEEN TOGGLED
                        <1> SCROLL_SHIFT equ    00010000b     ; SCROLL LOCK KEY IS DEPRESSED
                        <1> NUM_SHIFT    equ    00100000b     ; NUM LOCK KEY IS DEPRESSED
                        <1> CAPS_SHIFT   equ    01000000b     ; CAPS LOCK KEY IS DEPRE55ED
                        <1> INS_SHIFT    equ    10000000b     ; INSERT KEY IS DEPRESSED
                        <1> ;---------- FLAGS EQUATES WITHIN @KB_FLAG_2 -----------------------------------------
                        <1> KB_LEDS             equ    00000111b     ; KEYBOARD LED STATE BITS
                        <1> ;            equ    00000001b     ; SCROLL LOCK INDICATOR
                        <1> ;            equ    00000010b     ; NUM LOCK INDICATOR
                        <1> ;            equ    00000100b     ; CAPS LOCK INDICATOR
                        <1> ;            equ    00001000b     ; RESERVED (MUST BE ZERO)
                        <1> KB_FA        equ    00010000b     ; ACKNOWLEDGMENT RECEIVED
                        <1> KB_FE        equ    00100000b     ; RESEND RECEIVED FLAG
                        <1> KB_PR_LED    equ    01000000b     ; MODE INDICATOR UPDATE
                        <1> KB_ERR              equ    10000000b     ; KEYBOARD TRANSMIT ERROR FLAG
                        <1> ;----------- FLAGS EQUATES WITHIN @KB_FLAG_3 -----------------------------------
                        <1> LC_E1        equ    00000001b     ; LAST CODE WAS THE E1 HIDDEN CODE
                        <1> LC_E0        equ    00000010b     ; LAST CODE WAS THE E0 HIDDEN CODE
                        <1> R_CTL_SHIFT  equ    00000100b     ; RIGHT CTL KEY DOWN
                        <1> R_ALT_SHIFT  equ    00001000b     ; RIGHT ALT KEY DOWN
                        <1> GRAPH_ON     equ    00001000b     ; ALT GRAPHICS KEY DOWN (WT ONLY)
                        <1> KBX          equ    00010000b     ; ENHANCED KEYBOARD INSTALLED
                        <1> SET_NUM_LK   equ    00100000b     ; FORCE NUM LOCK IF READ ID AND KBX
                        <1> LC_AB        equ    01000000b     ; LAST CHARACTER WAS FIRST ID CHARACTER
                        <1> RD_ID        equ    10000000b     ; DOING A READ ID (MUST BE BIT0)
                        <1> ;
                        <1> ;----------- INTERRUPT EQUATES -------------------------------------------------
                        <1> EOI          equ    020h          ; END OF INTERRUPT COMMAND TO 8259
                        <1> INTA00              equ    020h          ; 8259 PORT
                        <1>
                        <1>
                        <1> kb_int:
                        <1>
                        <1> ; 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:
000010D6 FB             <1>      sti                     ; ENABLE INTERRUPTS
                        <1>      ;push  ebp
000010D7 50             <1>      push   eax
000010D8 53             <1>      push   ebx
000010D9 51             <1>      push   ecx
000010DA 52             <1>      push   edx
000010DB 56             <1>      push   esi
000010DC 57             <1>      push   edi
000010DD 1E             <1>      push   ds
000010DE 06             <1>      push   es
000010DF FC             <1>      cld                     ; FORWARD DIRECTION
000010E0 66B81000       <1>      mov    ax, KDATA
000010E4 8ED8           <1>      mov    ds, ax
000010E6 8EC0           <1>      mov    es, ax
                        <1>      ;
                        <1>      ;-----WAIT FOR KEYBOARD DISABLE COMMAND TO BE ACCEPTED
000010E8 B0AD           <1>      mov    al, DIS_KBD     ; DISABLE THE KEYBOARD COMMAND
```

```
000010EA E8A9050000          <1>       call   SHIP_IT                  ; EXECUTE DISABLE
000010EF FA                  <1>       cli                         ; DISABLE INTERRUPTS
000010F0 B900000100          <1>       mov    ecx, 10000h          ; SET MAXIMUM TIMEOUT
                             <1> KB_INT_01:
000010F5 E464                <1>       in     al, STATUS_PORT           ; READ ADAPTER STATUS
000010F7 A802                <1>       test   al, INPT_BUF_FULL    ; CHECK INPUT BUFFER FULL STATUS BIT
000010F9 E0FA                <1>       loopnz KB_INT_01           ; WAIT FOR COMMAND TO BE ACCEPTED
                             <1>       ;
                             <1>       ;----- READ CHARACTER FROM KEYBOARD INTERFACE
000010FB 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>       ;                          ; EXİT HANDLES HARDWARE EOI AND ENABLE
                             <1>       ;
                             <1>       ;----- CHECK FOR A RESEND COMMAND TO KEYBOARD
                             <1> KB_INT_02:                       ;        (AL)= SCAN CODE
000010FD FB                  <1>       sti                        ; ENABLE INTERRUPTS AGAIN
000010FE 3CFE                <1>       cmp    al, KB_RESEND       ; IS THE INPUT A RESEND
00001100 7411                <1>         je      short KB_INT_4          ; GO IF RESEND
                             <1>       ;
                             <1>       ;----- CHECK FOR RESPONSE TO A COMMAND TO KEYBOARD
00001102 3CFA                <1>       cmp    al, KB_ACK          ; IS THE INPUT AN ACKNOWLEDGE
00001104 751A                <1>         jne     short KB_INT_2          ; GO IF NOT
                             <1>       ;
                             <1>       ;----- A COMMAND TO THE KEYBOARD WAS ISSUED
00001106 FA                  <1>       cli                        ; DISABLE INTERRUPTS
00001107 800D[A76F0000]10    <1>       or     byte [KB_FLAG_2], KB_FA ; INDICATE ACK RECEIVED
0000110E E97A020000          <1>         jmp     K26                    ; RETURN IF NOT (ACK RETURNED FOR DATA)
                             <1>       ;
                             <1>       ;----- RESEND THE LAST BYTE
                             <1> KB_INT_4:
00001113 FA                  <1>       cli                        ; DISABLE INTERRUPTS
00001114 800D[A76F0000]20    <1>       or     byte [KB_FLAG_2], KB_FE ; INDICATE RESEND RECEIVED
0000111B E96D020000          <1>         jmp     K26                    ; RETURN IF NOT ACK RETURNED FOR DATA)
                             <1>       ;
                             <1> ;-----       UPDATE MODE INDICATORS IF CHANGE IN STATE
                             <1> KB_INT_2:
00001120 6650                <1>       push   ax                  ; SAVE DATA IN
00001122 E83D060000          <1>       call   MAKE_LED            ; GO GET MODE INDICATOR DATA BYTE
00001127 8A1D[A76F0000]      <1>       mov    bl, [KB_FLAG_2]     ; GET PREVIOUS BITS
0000112D 30C3                <1>       xor    bl, al              ; SEE IF ANY DIFFERENT
0000112F 80E307              <1>       and    bl, KB_LEDS         ; ISOLATE INDICATOR BITS
00001132 7405                <1>       jz     short UP0           ; IF NO CHANGE BYPASS UPDATE
00001134 E8C0050000          <1>       call   SND_LED                   ; GO TURN ON MODE INDICATORS
                             <1> UP0:
00001139 6658                <1>       pop    ax                  ; RESTORE DATA IN
                             <1> ;--------------------------------------------------------------------------
                             <1> ;     START OF KEY PROCESSING                                        ;
                             <1> ;--------------------------------------------------------------------------
0000113B 88C4                <1>       mov    ah, al              ; SAVE SCAN CODE IN AH ALSO
                             <1>       ;
                             <1>       ;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
0000113D 3CFF                <1>       cmp    al, KB_OVER_RUN           ; IS THIS AN OVERRUN CHAR
0000113F 0F843F050000        <1>         je      K62                     ; BUFFER_FULL_BEEP
                             <1>       ;
                             <1> K16:
00001145 8A3D[A86F0000]      <1>       mov    bh, [KB_FLAG_3]           ; LOAD FLAGS FOR TESTING
                             <1>       ;
                             <1>       ;----- TEST TO SEE IF A READ_ID IS IN PROGRESS
0000114B F6C7C0              <1>       test   bh, RD_ID+LC_AB     ; ARE WE DOING A READ ID?
0000114E 7449                <1>       jz     short NOT_ID        ; CONTINUE IF NOT
00001150 7917                <1>       jns    short TST_ID_2            ; IS THE RD_ID FLAG ON?
00001152 3CAB                <1>       cmp    al, ID_1            ; IS THIS THE 1ST ID CHARACTER?
00001154 7507                <1>       jne    short RST_RD_ID
00001156 800D[A86F0000]40    <1>       or     byte [KB_FLAG_3], LC_AB ; INDICATE 1ST ID WAS OK
                             <1> RST_RD_ID:
0000115D 8025[A86F0000]7F    <1>       and    byte [KB_FLAG_3], ~RD_ID ; RESET THE READ ID FLAG
                             <1>         ;jmp    short ID_EX            ; AND EXIT
00001164 E924020000          <1>       jmp    K26
                             <1>       ;
                             <1> TST_ID_2:
00001169 8025[A86F0000]BF    <1>       and    byte [KB_FLAG_3], ~LC_AB ; RESET FLAG
00001170 3C54                <1>       cmp    al, ID_2A           ; IS THIS THE 2ND ID CHARACTER?
00001172 7419                <1>         je     short KX_BIT        ; JUMP IF SO
00001174 3C41                <1>       cmp    al, ID_2            ; IS THIS THE 2ND ID CHARACTER?
                             <1>         ;jne   short ID_EX         ; LEAVE IF NOT
00001176 0F8511020000        <1>       jne    K26
                             <1>       ;
                             <1>       ;----- A READ ID SAID THAT IT WAS ENHANCED KEYBOARD
0000117C F6C720              <1>       test   bh, SET_NUM_LK             ; SHOULD WE SET NUM LOCK?
0000117F 740C                <1>         jz      short KX_BIT           ; EXIT IF NOT
00001181 800D[A56F0000]20    <1>       or     byte [KB_FLAG], NUM_STATE ; FORCE NUM LOCK ON
00001188 E86C050000          <1>       call   SND_LED                   ; GO SET THE NUM LOCK INDICATOR
                             <1> KX_BIT:
0000118D 800D[A86F0000]10    <1>       or     byte [KB_FLAG_3], KBX     ; INDICATE ENHANCED KEYBOARD WAS FOUND
00001194 E9F4010000          <1> ID_EX:       jmp     K26               ; EXIT
                             <1>       ;
                             <1> NOT_ID:
00001199 3CE0                <1>       cmp    al, MC_E0           ; IS THIS THE GENERAL MARKER CODE?
0000119B 750C                <1>       jne    short TEST_E1
0000119D 800D[A86F0000]12    <1>       or     byte [KB_FLAG_3], LC_E0+KBX ; SET FLAG BIT, SET KBX, AND
                             <1>       ;jmp   short EXIT          ; THROW AWAY THIS CODE
000011A4 E9EB010000          <1>       jmp    K26A
                             <1> TEST_E1:
000011A9 3CE1                <1>       cmp    al, MC_E1           ; IS THIS THE PAUSE KEY?
000011AB 750C                <1>       jne    short NOT_HC
000011AD 800D[A86F0000]11    <1>       or     byte [KB_FLAG_3], LC_E1+KBX ; SET FLAG BIT, SET KBX, AND
000011B4 E9DB010000          <1> EXIT: jmp    K26A                ; THROW AWAY THIS CODE
                             <1>       ;
```

```
                                <1> NOT_HC:
000011B9 247F                   <1>       and    al, 07Fh           ; TURN OFF THE BREAK BIT
000011BB F6C702                 <1>       test   bh, LC_E0          ; LAST CODE THE E0 MARKER CODE
000011BE 7414                   <1>       jz     short NOT_LC_E0          ; JUMP IF NOT
                                <1>       ;
000011C0 BF[926E0000]           <1>       mov    edi, _K6+6         ; IS THIS A SHIFT KEY?
000011C5 AE                     <1>       scasb
000011C6 0F84C1010000           <1>         je      K26 ; K16B              ; YES, THROW AWAY & RESET FLAG
000011CC AE                     <1>       scasb
000011CD 757C                   <1>       jne    short K16A         ; NO, CONTINUE KEY PROCESSING
                                <1>       ;jmp   short K16B         ; YES, THROW AWAY & RESET FLAG
000011CF E9B9010000             <1>       jmp    K26
                                <1>       ;
                                <1> NOT_LC_E0:
000011D4 F6C701                 <1>       test   bh, LC_E1          ; LAST CODE THE E1 MARKER CODE?
000011D7 7435                   <1>       jz     short T_SYS_KEY          ; JUMP IF NOT
000011D9 B904000000             <1>       mov    ecx, 4             ; LENGHT OF SEARCH
000011DE BF[906E0000]           <1>       mov    edi, _K6+4         ; IS THIS AN ALT, CTL, OR SHIFT?
000011E3 F2AE                   <1>       repne  scasb              ; CHECK IT
                                <1>       ;je    short EXIT         ; THROW AWAY IF SO
000011E5 0F84A9010000           <1>       je     K26A
                                <1>       ;
000011EB 3C45                   <1>       cmp    al, NUM_KEY        ; IS IT THE PAUSE KEY?
                                <1>       ;jne   short K16B         ; NO, THROW AWAY & RESET FLAG
000011ED 0F859A010000           <1>       jne    K26
000011F3 F6C480                 <1>       test   ah, 80h                   ; YES, IS IT THE BREAK OF THE KEY?
                                <1>       ;jnz   short K16B         ; YES, THROW THIS AWAY, TOO
000011F6 0F8591010000           <1>       jnz    K26
                                <1>         ; 20/02/2015
000011FC F605[A66F0000]08       <1>       test   byte [KB_FLAG_1],HOLD_STATE ;  NO, ARE WE PAUSED ALREADY?
                                <1>       ;jnz   short K16B         ;  YES, THROW AWAY
00001203 0F8584010000           <1>       jnz    K26
00001209 E9E1020000             <1>       jmp     K39P                    ; NO, THIS IS THE REAL PAUSE STATE
                                <1>       ;
                                <1>       ;----- TEST FOR SYSTEM KEY
                                <1> T_SYS_KEY:
0000120E 3C54                   <1>       cmp    al, SYS_KEY        ; IS IT THE SYSTEM KEY?
00001210 7539                   <1>       jnz    short K16A         ; CONTINUE IF NOT
                                <1>       ;
00001212 F6C480                 <1>       test   ah, 80h                   ; CHECK IF THIS A BREAK CODE
00001215 7524                   <1>       jnz    short K16C         ; DO NOT TOUCH SYSTEM INDICATOR IF TRUE
                                <1>       ;
00001217 F605[A66F0000]04       <1>       test   byte [KB_FLAG_1], SYS_SHIFT ; SEE IF IN SYSTEM KEY HELD DOWN
                                <1>         ;jnz short K16B         ; IF YES, DO NOT PROCESS SYSTEM INDICATOR
0000121E 0F8569010000           <1>       jnz     K26
                                <1>       ;
00001224 800D[A66F0000]04       <1>       or     byte [KB_FLAG_1], SYS_SHIFT ; INDICATE SYSTEM KEY DEPRESSED
0000122B B020                   <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
0000122D E620                   <1>       out    20h, al ;out INTA00, al    ; SEND COMMAND TO INTERRUPT CONTROL PORT
                                <1>                                      ; INTERRUPT-RETURN-NO-EOI
0000122F B0AE                   <1>       mov    al, ENA_KBD        ; INSURE KEYBOARD IS ENABLED
00001231 E862040000             <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
00001236 E965010000             <1>         jmp     K27A                    ; END PROCESSING
                                <1>       ;
                                <1> ;K16B:       jmp    K26                ; IGNORE SYSTEM KEY
                                <1>       ;
                                <1> K16C:
0000123B 8025[A66F0000]FB       <1>       and    byte [KB_FLAG_1], ~SYS_SHIFT ; TURN OFF SHIFT KEY HELD DOWN
00001242 B020                   <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
00001244 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>       ;
00001246 E94E010000             <1>       jmp     K27               ; IGNORE SYSTEM KEY
                                <1>       ;
                                <1>       ;----- TEST FOR SHIFT KEYS
                                <1> K16A:
0000124B 8A1D[A56F0000]         <1>       mov    bl, [KB_FLAG]      ; PUT STATE FLAGS IN BL
00001251 BF[8C6E0000]           <1>       mov    edi, _K6           ; SHIFT KEY TABLE offset
00001256 B908000000             <1>       mov    ecx, _K6L          ; LENGTH
0000125B F2AE                   <1>       repne  scasb              ; LOOK THROUGH THE TABLE FOR A MATCH
0000125D 88E0                   <1>       mov    al, ah             ; RECOVER SCAN CODE
0000125F 0F8510010000           <1>         jne     K25                     ; IF NO MATCH, THEN SHIFT NOT FOUND
                                <1>       ;
                                <1>       ;------      SHIFT KEY FOUND
                                <1> K17:
00001265 81EF[8D6E0000]         <1>         sub     edi, _K6+1              ; ADJUST PTR TO SCAN CODE MATCH
0000126B 8AA7[946E0000]         <1>             mov     ah, [edi+_K7]           ; GET MASK INTO AH
00001271 B102                   <1>       mov    cl, 2              ; SETUP COUNT FOR FLAG SHIFTS
00001273 A880                   <1>       test   al, 80h                  ; TEST FOR BREAK KEY
00001275 0F8596000000           <1>         jnz     K23                     ; JUMP OF BREAK
                                <1>       ;
                                <1>       ;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
                                <1> K17C:
0000127B 80FC10                 <1>       cmp    ah, SCROLL_SHIFT
0000127E 732B                   <1>       jae    short K18          ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
                                <1>       ;
                                <1>       ;----- PLAIN SHIFT KEY, SET SHIFT ON
00001280 0825[A56F0000]         <1>       or     [KB_FLAG], ah      ; TURN ON SHIFT BIT
00001286 A80C                   <1>         test al, CTL_SHIFT+ALT_SHIFT ; IS IT ALT OR CTRL?
                                <1>       ;jnz   short K17D         ; YES, MORE FLAGS TO SET
00001288 0F84FF000000           <1>       jz     K26                ; NO, INTERRUPT RETURN
                                <1> K17D:
0000128E F6C702                 <1>       test   bh, LC_E0          ; IS THIS ONE OF NEW KEYS?
00001291 740B                   <1>       jz     short K17E         ; NO, JUMP
```

```
00001293 0825[A86F0000]      <1>       or    [KB_FLAG_3], ah           ; SET BITS FOR RIGHT CTRL, ALT
00001299 E9EF000000          <1>       jmp   K26                 ; INTERRUPT RETURN
                             <1> K17E:
0000129E D2EC                <1>       shr   ah, cl              ; MOVE FLAG BITS TWO POSITIONS
000012A0 0825[A66F0000]      <1>       or    [KB_FLAG_1], ah           ; SET BITS FOR LEFT CTRL, ALT
000012A6 E9E2000000          <1>       jmp   K26
                             <1>       ;
                             <1>       ;-----TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
                             <1> K18:                            ; SHIFT-TOGGLE
000012AB F6C304              <1>       test  bl, CTL_SHIFT             ; CHECK CTL SHIFT STATE
                             <1>         ;jz       short K18A              ; JUMP IF NOT CTL STATE
000012AE 0F85C1000000        <1>         jnz     K25                     ; JUMP IF CTL STATE
                             <1> K18A:
000012B4 3C52                <1>       cmp   al, INS_KEY         ; CHECK FOR INSERT KEY
000012B6 7524                <1>       jne   short K22           ; JUMP IF NOT INSERT KEY
000012B8 F6C308              <1>       test  bl, ALT_SHIFT             ; CHECK FOR ALTERNATE SHIFT
                             <1>             ;jz    short K18B         ; JUMP IF NOT ALTERNATE SHIFT
000012BB 0F85B4000000        <1>         jnz     K25                     ; JUMP IF ALTERNATE SHIFT
                             <1> K18B:
000012C1 F6C702              <1>       test  bh, LC_E0 ;20/02/2015     ; IS THIS NEW INSERT KEY?
000012C4 7516                <1>       jnz   short K22           ; YES, THIS ONE'S NEVER A '0'
                             <1> K19:
000012C6 F6C320              <1>       test  bl, NUM_STATE             ; CHECK FOR BASE STATE
000012C9 750C                <1>       jnz   short K21           ; JUMP IF NUM LOCK IS ON
000012CB F6C303              <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST FOR SHIFT STATE
000012CE 740C                <1>       jz    short K22           ; JUMP IF BASE STATE
                             <1> K20:                            ; NUMERIC ZERO, NOT INSERT KEY
000012D0 88C4                <1>       mov   ah, al              ; PUT SCAN CODE BACK IN AH
000012D2 E99E000000          <1>         jmp     K25                     ; NUMERAL '0', STNDRD. PROCESSING
                             <1> K21:                            ; MIGHT BE NUMERIC
000012D7 F6C303              <1>       test  bl, LEFT_SHIFT+RIGHT_SHIFT
000012DA 74F4                <1>       jz    short K20           ; IS NUMERIC, STD. PROC.
                             <1>       ;
                             <1> K22:                            ; SHIFT TOGGLE KEY HIT; PROCESS IT
000012DC 8425[A66F0000]      <1>       test  ah, [KB_FLAG_1]     ; IS KEY ALREADY DEPRESSED
000012E2 0F85A5000000        <1>         jnz     K26                     ; JUMP IF KEY ALREADY DEPRESSED
                             <1> K22A:
000012E8 0825[A66F0000]      <1>         or      [KB_FLAG_1], ah ; INDICATE THAT THE KEY IS DEPRESSED
000012EE 3025[A56F0000]      <1>       xor   [KB_FLAG], ah       ; TOGGLE THE SHIFT STATE
                             <1>       ;
                             <1>       ;-----TOGGLE LED IF CAPS, NUM  OR SCROLL KEY DEPRESSED
000012F4 F6C470              <1>       test  ah, CAPS_SHIFT+NUM_SHIFT+SCROLL_SHIFT ; SHIFT TOGGLE?
000012F7 7409                <1>       jz    short K22B          ; GO IF NOT
                             <1>       ;
000012F9 6650                <1>       push  ax                  ; SAVE SCAN CODE AND SHIFT MASK
000012FB E8F9030000          <1>       call  SND_LED                   ; GO TURN MODE INDICATORS ON
00001300 6658                <1>       pop   ax                  ; RESTORE SCAN CODE
                             <1> K22B:
00001302 3C52                <1>       cmp   al, INS_KEY         ; TEST FOR 1ST MAKE OF INSERT KEY
00001304 0F8583000000        <1>         jne     K26                     ; JUMP IF NOT INSERT KEY
0000130A 88C4                <1>       mov   ah, al              ; SCAN CODE IN BOTH HALVES OF AX
0000130C E999000000          <1>         jmp     K28                     ; FLAGS UPDATED, PROC. FOR BUFFER
                             <1>       ;
                             <1>       ;-----BREAK SHIFT FOUND
                             <1> K23:                            ; BREAK-SHIFT-FOUND
00001311 80FC10              <1>       cmp   ah, SCROLL_SHIFT    ; IS THIS A TOGGLE KEY
00001314 F6D4                <1>       not   ah                  ; INVERT MASK
00001316 7355                <1>       jae   short K24           ; YES, HANDLE BREAK TOGGLE
00001318 2025[A56F0000]      <1>       and   [KB_FLAG], ah       ; TURN OFF SHIFT BIT
0000131E 80FCFB              <1>       cmp   ah, ~CTL_SHIFT            ; IS THIS ALT OR CTL?
00001321 7730                <1>       ja    short K23D          ; NO, ALL DONE
                             <1>       ;
00001323 F6C702              <1>       test  bh, LC_E0           ; 2ND ALT OR CTL?
00001326 7408                <1>       jz    short K23A          ; NO, HANSLE NORMALLY
00001328 2025[A86F0000]      <1>       and   [KB_FLAG_3], ah           ; RESET BIT FOR RIGHT ALT OR CTL
0000132E EB08                <1>       jmp   short K23B          ; CONTINUE
                             <1> K23A:
00001330 D2FC                <1>       sar   ah, cl              ; MOVE THE MASK BIT TWO POSITIONS
00001332 2025[A66F0000]      <1>       and   [KB_FLAG_1], ah           ; RESET BIT FOR LEFT ALT AND CTL
                             <1> K23B:
00001338 88C4                <1>       mov   ah, al              ; SAVE SCAN CODE
0000133A A0[A86F0000]        <1>       mov   al, [KB_FLAG_3]           ; GET RIGHT ALT & CTRL FLAGS
0000133F D2E8                <1>       shr   al, cl              ; MOVE TO BITS 1 & 0
00001341 0A05[A66F0000]      <1>       or    al, [KB_FLAG_1]           ; PUT IN LEFT ALŞT & CTL FLAGS
00001347 D2E0                <1>       shl   al, cl              ; MOVE BACK TO BITS 3 & 2
00001349 240C                <1>       and   al, ALT_SHIFT+CTL_SHIFT ; FILTER OUT OTHER GARBAGE
0000134B 0805[A56F0000]      <1>       or    [KB_FLAG], al       ; PUT RESULT IN THE REAL FLAGS
00001351 88E0                <1>       mov   al, ah
                             <1> K23D:
00001353 3CB8                <1>       cmp   al, ALT_KEY+80h           ; IS THIS ALTERNATE SHIFT RELEASE
00001355 7536                <1>       jne   short K26           ; INTERRUPT RETURN
                             <1>       ;
                             <1>       ;-----ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
00001357 A0[A96F0000]        <1>       mov   al, [ALT_INPUT]
0000135C B400                <1>       mov   ah, 0               ; SCAN CODE OF 0
0000135E 8825[A96F0000]      <1>       mov   [ALT_INPUT], ah     ; ZERO OUT THE FIELD
00001364 3C00                <1>       cmp   al, 0               ; WAS THE INPUT = 0?
00001366 7425                <1>       je    short K26           ; INTERRUPT_RETURN
                             <1>         ; 29/01/2016
                             <1>       ;jmp    K61                     ; IT WASN'T, SO PUT IN BUFFER
00001368 E9D0020000          <1>       jmp   _K60
                             <1>       ;
                             <1> K24:                            ; BREAK-TOGGLE
0000136D 2025[A66F0000]      <1>       and   [KB_FLAG_1], ah     ; INDICATE NO LONGER DEPRESSED
00001373 EB18                <1>       jmp   short K26           ; INTERRUPT_RETURN
                             <1>       ;
                             <1>       ;-----TEST FOR HOLD STATE
                             <1>                                 ; AL, AH = SCAN CODE
                             <1> K25:                            ; NO-SHIFT-FOUND
00001375 3C80                <1>       cmp   al, 80h                   ; TEST FOR BREAK KEY
00001377 7314                <1>       jae   short K26           ; NOTHING FOR BREAK CHARS FROM HERE ON
00001379 F605[A66F0000]08    <1>       test  byte [KB_FLAG_1], HOLD_STATE ; ARE WE IN HOLD STATE
00001380 7428                <1>       jz    short K28           ; BRANCH AROUND TEST IF NOT
00001382 3C45                <1>       cmp   al, NUM_KEY
00001384 7407                <1>       je    short K26           ; CAN'T END HOLD ON NUM_LOCK
```

```
00001386 8025[A66F0000]F7    <1>       and   byte [KB_FLAG_1], ~HOLD_STATE ; TURN OFF THE HOLD STATE BIT
                             <1>       ;
                             <1> K26:
0000138D 8025[A86F0000]FC    <1>       and   byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; RESET LAST CHAR H.C. FLAG
                             <1> K26A:                         ; INTERRUPT-RETURN
00001394 FA                  <1>       cli                     ; TURN OFF INTERRUPTS
00001395 B020                <1>       mov   al, EOI                 ; END OF INTERRUPT COMMAND
00001397 E620                <1>       out   20h, al      ;out INTA00, al     ; SEND COMMAND TO INTERRUPT CONTROL PORT
                             <1> K27:                          ; INTERRUPT-RETURN-NO-EOI
00001399 B0AE                <1>       mov   al, ENA_KBD       ; INSURE KEYBOARD IS ENABLED
0000139B E8F8020000          <1>       call  SHIP_IT                 ; EXECUTE ENABLE
                             <1> K27A:
000013A0 FA                  <1>       cli                     ; DISABLE INTERRUPTS
                             <1>       ;;mov byte [intflg], 0 ; 07/01/2017 ;; 15/01/2017
000013A1 07                  <1>       pop   es                ; RESTORE REGISTERS
000013A2 1F                  <1>       pop   ds
000013A3 5F                  <1>       pop   edi
000013A4 5E                  <1>       pop   esi
000013A5 5A                  <1>       pop   edx
000013A6 59                  <1>       pop   ecx
000013A7 5B                  <1>       pop   ebx
000013A8 58                  <1>       pop   eax
                             <1>       ;pop  ebp
000013A9 CF                  <1>       iretd                   ; RETURN
                             <1>
                             <1>       ;----- NOT IN HOLD STATE
                             <1> K28:                          ; NO-HOLD-STATE
000013AA 3C58                <1>       cmp   al, 88            ; TEST FOR OUT-OF-RANGE SCAN CODES
000013AC 77DF                <1>       ja    short K26         ; IGNORE IF OUT-OF-RANGE
                             <1>       ;
000013AE F6C308              <1>       test  bl, ALT_SHIFT           ; ARE WE IN ALTERNATE SHIFT
                             <1>         ;jz short K28A        ; IF NOT ALTERNATE
000013B1 0F84F1000000        <1>         jz    K38
                             <1>       ;
000013B7 F6C710              <1>       test  bh, KBX                 ; IS THIS THE ENCHANCED KEYBOARD?
000013BA 740D                <1>       jz    short K29         ; NO, ALT STATE IS REAL
                             <1>        ;28/02/2015
000013BC F605[A66F0000]04    <1>       test  byte [KB_FLAG_1], SYS_SHIFT ; YES, IS SYSREQ KEY DOWN?
                             <1>       ;jz   short K29         ;  NO, ALT STATE IS REAL
000013C3 0F85DF000000        <1>       jnz   K38               ; YES, THIS IS PHONY ALT STATE
                             <1>         ;                     ; DUE TO PRESSING SYSREQ
                             <1> ;K28A:      jmp   short K38
                             <1>       ;
                             <1>       ;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
                             <1> K29:                          ; TEST-RESET
000013C9 F6C304              <1>       test  bl, CTL_SHIFT           ; ARE WE IN CONTROL SHIFT ALSO?
000013CC 740B                <1>       jz    short K31         ; NO_RESET
000013CE 3C53                <1>       cmp   al, DEL_KEY       ; CTL-ALT STATE, TEST FOR DELETE KEY
000013D0 7507                <1>       jne   short K31         ; NO_RESET, IGNORE
                             <1>       ;
                             <1>       ;----- CTL-ALT-DEL HAS BEEN FOUND
                             <1>       ; 26/08/2014
                             <1> cpu_reset:
                             <1>       ; IBM PC/AT ROM BIOS source code - 10/06/85 (TEST4.ASM - PROC_SHUTDOWN)
                             <1>       ; Send FEh (system reset command) to the keyboard controller.
000013D2 B0FE                <1>       mov   al, SHUT_CMD      ; SHUTDOWN COMMAND
000013D4 E664                <1>       out   STATUS_PORT, al         ; SEND TO KEYBOARD CONTROL PORT
                             <1> khere:
000013D6 F4                  <1>       hlt                     ; WAIT FOR 80286 RESET
000013D7 EBFD                <1>       jmp   short khere       ; INSURE HALT
                             <1>
                             <1>       ;
                             <1>       ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
                             <1> K31:                          ; NO-RESET
000013D9 3C39                <1>       cmp   al, 57            ; TEST FOR SPACE KEY
000013DB 7507                <1>       jne   short K311        ; NOT THERE
000013DD B020                <1>       mov   al, ' '                 ; SET SPACE CHAR
000013DF E948020000          <1>         jmp     K57                  ; BUFFER_FILL
                             <1> K311:
000013E4 3C0F                <1>       cmp   al, 15            ; TEST FOR TAB KEY
000013E6 7509                <1>       jne   short K312        ; NOT THERE
000013E8 66B800A5            <1>       mov   ax, 0A500h        ; SET SPECIAL CODE FOR ALT-TAB
000013EC E93B020000          <1>         jmp     K57                  ; BUFFER_FILL
                             <1> K312:
000013F1 3C4A                <1>       cmp   al, 74            ; TEST FOR KEY PAD -
000013F3 0F84A2000000        <1>         je      K37B                ; GO PROCESS
000013F9 3C4E                <1>       cmp   al, 78            ; TEST FOR KEY PAD +
000013FB 0F849A000000        <1>         je      K37B                ; GO PROCESS
                             <1>       ;
                             <1>       ;----- LOOK FOR KEY PAD ENTRY
                             <1> K32:                          ; ALT-KEY-PAD
00001401 BF[686E0000]        <1>       mov   edi, K30          ; ALT-INPUT-TABLE offset
00001406 B90A000000          <1>       mov   ecx, 10                ; LOOK FOR ENTRY USING KEYPAD
0000140B F2AE                <1>       repne scasb             ; LOOK FOR MATCH
0000140D 7525                <1>       jne   short K33         ; NO_ALT_KEYPAD
0000140F F6C702              <1>       test  bh, LC_E0         ; IS THIS ONE OF THE NEW KEYS?
00001412 0F858A000000        <1>         jnz     K37C                ; YES, JUMP, NOT NUMPAD KEY
00001418 81EF[696E0000]      <1>       sub   edi, K30+1        ; DI NOW HAS ENTRY VALUE
0000141E A0[A96F0000]        <1>       mov   al, [ALT_INPUT]   ; GET THE CURRENT BYTE
00001423 B40A                <1>       mov   ah, 10            ; MULTIPLY BY 10
00001425 F6E4                <1>       mul   ah
00001427 6601F8              <1>       add   ax, di            ; ADD IN THE LATEST ENTRY
0000142A A2[A96F0000]        <1>       mov   [ALT_INPUT], al   ; STORE IT AWAY
                             <1> ;K32A:
0000142F E959FFFFFF          <1>         jmp     K26                  ; THROW AWAY THAT KEYSTROKE
                             <1>       ;
                             <1>       ;----- LOOK FOR SUPERSHIFT ENTRY
                             <1> K33:                          ; NO-ALT-KEYPAD
00001434 C605[A96F0000]00    <1>         mov     byte [ALT_INPUT], 0   ; ZERO ANY PREVIOUS ENTRY INTO INPUT
0000143B B91A000000          <1>       mov   ecx, 26                ; (DI),(ES) ALREADY POINTING
00001440 F2AE                <1>       repne scasb             ; LOOK FOR MATCH IN ALPHABET
00001442 7450                <1>       je    short K37A        ; MATCH FOUND, GO FILLL THE BUFFER
                             <1>       ;
                             <1>       ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT
                             <1> K34:                          ; ALT-TOP-ROW
```

```
 985 00001444 3C02              <1>       cmp    al, 2               ; KEY WITH '1' ON IT
 986 00001446 7253              <1>       jb     short K37B          ; MUST BE ESCAPE
 987 00001448 3C0D              <1>       cmp    al, 13              ; IS IT IN THE REGION
 988 0000144A 7705              <1>       ja     short K35           ; NO, ALT SOMETHING ELSE
 989 0000144C 80C476            <1>       add    ah, 118                  ; CONVERT PSEUDO SCAN CODE TO RANGE
 990 0000144F EB43              <1>       jmp    short K37A          ; GO FILL THE BUFFER
 991                            <1>       ;
 992                            <1>       ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
 993                            <1> K35:                             ; ALT-FUNCTION
 994 00001451 3C57              <1>       cmp    al, F11_M           ; IS IT F11?
 995 00001453 7209              <1>       jb     short K35A ; 20/02/2015    ; NO, BRANCH
 996 00001455 3C58              <1>       cmp    al, F12_M           ; IS IT F12?
 997 00001457 7705              <1>       ja     short K35A ; 20/02/2015    ; NO, BRANCH
 998 00001459 80C434            <1>       add    ah, 52              ; CONVERT TO PSEUDO SCAN CODE
 999 0000145C EB36              <1>       jmp    short K37A          ; GO FILL THE BUFFER
1000                            <1> K35A:
1001 0000145E F6C702            <1>       test   bh, LC_E0           ; DO WE HAVE ONE OF THE NEW KEYS?
1002 00001461 7422              <1>       jz     short K37           ; NO, JUMP
1003 00001463 3C1C              <1>       cmp    al, 28              ; TEST FOR KEYPAD ENTER
1004 00001465 7509              <1>         jne     short K35B              ; NOT THERE
1005 00001467 66B800A6          <1>       mov    ax, 0A600h          ; SPECIAL CODE
1006 0000146B E9BC010000        <1>       jmp    K57                 ; BUFFER FILL
1007                            <1> K35B:
1008 00001470 3C53              <1>       cmp    al, 83              ; TEST FOR DELETE KEY
1009 00001472 742E              <1>       je     short K37C          ; HANDLE WITH OTHER EDIT KEYS
1010 00001474 3C35              <1>       cmp    al, 53              ; TEST FOR KEYPAD /
1011                            <1>       ;jne   short K32A          ; NOT THERE, NO OTHER E0 SPECIALS
1012 00001476 0F8511FFFFFF      <1>         jne     K26
1013 0000147C 66B800A4          <1>       mov    ax, 0A400h          ; SPECIAL CODE
1014 00001480 E9A7010000        <1>       jmp    K57                 ; BUFFER FILL
1015                            <1> K37:
1016 00001485 3C3B              <1>       cmp    al, 59              ; TEST FOR FUNCTION KEYS (F1)
1017 00001487 7212              <1>         jb      short K37B              ; NO FN, HANDLE W/OTHER EXTENDED
1018 00001489 3C44              <1>       cmp    al, 68              ; IN KEYPAD REGION?
1019                            <1>         ;ja  short K32A          ; IF SO, IGNORE
1020 0000148B 0F87FCFEFFFF      <1>         ja      K26
1021 00001491 80C42D            <1>       add    ah, 45              ; CONVERT TO PSEUDO SCAN CODE
1022                            <1> K37A:
1023 00001494 B000              <1>       mov    al, 0               ; ASCII CODE OF ZERO
1024 00001496 E991010000        <1>         jmp     K57                     ; PUT IT IN THE BUFFER
1025                            <1> K37B:
1026 0000149B B0F0              <1>       mov    al, 0F0h            ; USE SPECIAL ASCII CODE
1027 0000149D E98A010000        <1>       jmp     K57                     ; PUT IT IN THE BUFFER
1028                            <1> K37C:
1029 000014A2 0450              <1>       add    al, 80              ; CONVERT SCAN CODE (EDIT KEYS)
1030 000014A4 88C4              <1>       mov    ah, al              ; (SCAN CODE NOT IN AH FOR INSERT)
1031 000014A6 EBEC              <1>       jmp     short K37A              ; PUT IT IN THE BUFFER
1032                            <1>       ;
1033                            <1>       ;----- NOT IN ALTERNATE SHIFT
1034                            <1> K38:                             ; NOT-ALT-SHIFT
1035                            <1>                                  ; BL STILL HAS SHIFT FLAGS
1036 000014A8 F6C304            <1>       test   bl, CTL_SHIFT             ; ARE WE IN CONTROL SHIFT?
1037                            <1>       ;jnz   short K38A          ; YES, START PROCESSING
1038 000014AB 0F84B0000000      <1>         jz      K44                      ; NOT-CTL-SHIFT
1039                            <1>       ;
1040                            <1>       ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
1041                            <1>       ;----- TEST FOR BREAK
1042                            <1> K38A:
1043 000014B1 3C46              <1>       cmp    al, SCROLL_KEY            ; TEST FOR BREAK
1044 000014B3 7531              <1>       jne    short K39           ; JUMP, NO-BREAK
1045 000014B5 F6C710            <1>       test   bh, KBX                   ; IS THIS THE ENHANCED KEYBOARD?
1046 000014B8 7405              <1>       jz     short K38B          ; NO, BREAK IS VALID
1047 000014BA F6C702            <1>       test   bh, LC_E0           ; YES, WAS LAST CODE AN E0?
1048 000014BD 7427              <1>       jz     short K39           ; NO-BREAK, TEST FOR PAUSE
1049                            <1> K38B:
1050 000014BF 8B1D[B26F0000]    <1>       mov    ebx, [BUFFER_HEAD]  ; RESET BUFFER TO EMPTY
1051 000014C5 891D[B66F0000]    <1>       mov    [BUFFER_TAIL], ebx
1052 000014CB C605[A46F0000]80  <1>       mov    byte [BIOS_BREAK], 80h  ; TURN ON BIOS_BREAK BIT
1053                            <1>       ;
1054                            <1>       ;----- ENABLE KEYBOARD
1055 000014D2 B0AE              <1>       mov    al, ENA_KBD         ; ENABLE KEYBOARD
1056 000014D4 E8BF010000        <1>       call   SHIP_IT                  ; EXECUTE ENABLE
1057                            <1>       ;
1058                            <1>       ; CTRL+BREAK code here !!!
1059                            <1>       ;INT   1BH                 ; BREAK INTERRUPT VECTOR
1060                            <1>       ; 17/10/2015
1061 000014D9 E812610000        <1>       call   ctrlbrk ; control+break subroutine
1062                            <1>       ;
1063 000014DE 6629C0            <1>       sub    ax, ax              ; PUT OUT DUMMY CHARACTER
1064 000014E1 E946010000        <1>         jmp     K57                     ; BUFFER_FILL
1065                            <1>       ;
1066                            <1>       ;----- TEST FOR PAUSE
1067                            <1> K39:                             ; NO_BREAK
1068 000014E6 F6C710            <1>       test   bh, KBX                   ; IS THIS THE ENHANCED KEYBOARD?
1069 000014E9 7537              <1>       jnz    short K41           ; YES, THEN THIS CAN'T BE PAUSE
1070 000014EB 3C45              <1>       cmp    al, NUM_KEY         ; LOOK FOR PAUSE KEY
1071 000014ED 7533              <1>       jne    short K41           ; NO-PAUSE
1072                            <1> K39P:
1073 000014EF 800D[A66F0000]08  <1>       or     byte [KB_FLAG_1], HOLD_STATE ; TURN ON THE HOLD FLAG
1074                            <1>       ;
1075                            <1>       ;----- ENABLE KEYBOARD
1076 000014F6 B0AE              <1>       mov    al, ENA_KBD         ; ENABLE KEYBOARD
1077 000014F8 E89B010000        <1>       call   SHIP_IT                  ; EXECUTE ENABLE
1078                            <1> K39A:
1079 000014FD B020              <1>       mov    al, EOI                  ; END OF INTERRUPT TO CONTROL PORT
1080 000014FF E620              <1>       out    20h, al ;out INTA00, al   ; ALLOW FURTHER KEYSTROKE INTERRUPTS
1081                            <1>       ;
1082                            <1>       ;----- DURING PAUSE INTERVAL, TURN COLOR CRT BACK ON
1083 00001501 803D[DA6F0000]07  <1>         cmp     byte [CRT_MODE], 7      ; IS THIS BLACK AND WHITE CARD
1084 00001508 740A              <1>         je      short K40                   ; YES, NOTHING TO DO
1085 0000150A 66BAD803          <1>       mov    dx, 03D8h           ; PORT FOR COLOR CARD
1086 0000150E A0[DB6F0000]      <1>         mov     al, [CRT_MODE_SET]     ; GET THE VALUE OF THE CURRENT MODE
1087 00001513 EE                <1>       out    dx, al              ; SET THE CRT MODE, SO THAT CRT IS ON
1088                            <1>       ;
1089                            <1> K40:                             ; PAUSE-LOOP
```

```
1090 00001514 F605[A66F0000]08    <1>         test    byte [KB_FLAG_1], HOLD_STATE ; CHECK HOLD STATE FLAG
1091 0000151B 75F7                <1>       jnz    short K40           ; LOOP UNTIL FLAG TURNED OFF
1092                              <1>       ;
1093 0000151D E977FEFFFF          <1>         jmp     K27                     ; INTERRUPT_RETURN_NO_EOI
1094                              <1>         ;
1095                              <1>       ;----- TEST SPECIAL CASE KEY 55
1096                              <1> K41:                              ; NO-PAUSE
1097 00001522 3C37                <1>       cmp    al, 55               ; TEST FOR */PRTSC KEY
1098 00001524 7513                <1>       jne    short K42           ; NOT-KEY-55
1099 00001526 F6C710              <1>       test   bh, KBX                    ; IS THIS THE ENHANCED KEYBOARD?
1100 00001529 7405                <1>       jz     short K41A          ; NO, CTL-PRTSC IS VALID
1101 0000152B F6C702              <1>       test   bh, LC_E0           ; YES, WAS LAST CODE AN E0?
1102 0000152E 7421                <1>       jz     short K42B          ; NO, TRANSLATE TO A FUNCTION
1103                              <1> K41A:
1104 00001530 66B80072            <1>       mov    ax, 114*256         ; START/STOP PRINTING SWITCH
1105 00001534 E9F3000000          <1>         jmp     K57                     ; BUFFER_FILL
1106                              <1>       ;
1107                              <1>       ;----- SET UP TO TRANSLATE CONTROL SHIFT
1108                              <1> K42:                              ; NOT-KEY-55
1109 00001539 3C0F                <1>       cmp    al, 15               ; IS IT THE TAB KEY?
1110 0000153B 7414                <1>       je     short K42B          ; YES, XLATE TO FUNCTION CODE
1111 0000153D 3C35                <1>       cmp    al, 53               ; IS IT THE / KEY?
1112 0000153F 750E                <1>       jne    short K42A          ; NO, NO MORE SPECIAL CASES
1113 00001541 F6C702              <1>       test   bh, LC_E0           ; YES, IS IT FROM THE KEY PAD?
1114 00001544 7409                <1>       jz     short K42A          ; NO, JUST TRANSLATE
1115 00001546 66B80095            <1>       mov    ax, 9500h           ; YES, SPECIAL CODE FOR THIS ONE
1116 0000154A E9DD000000          <1>       jmp    K57                 ; BUFFER FILL
1117                              <1> K42A:
1118                              <1>       ;;mov  ebx, _K8            ; SET UP TO TRANSLATE CTL
1119 0000154F 3C3B                <1>       cmp    al, 59               ; IS IT IN CHARACTER TABLE?
1120                              <1>         ;jb  short K45F               ; YES, GO TRANSLATE CHAR
1121                              <1>       ;;jb   K56 ; 20/02/2015
1122                              <1>       ;;jmp  K64 ; 20/02/2015
1123                              <1> K42B:
1124 00001551 BB[9C6E0000]        <1>       mov    ebx, _K8             ; SET UP TO TRANSLATE CTL
1125 00001556 0F82AE000000        <1>       jb     K56 ;; 20/02/2015
1126 0000155C E9B9000000          <1>       jmp    K64
1127                              <1>         ;
1128                              <1>       ;----- NOT IN CONTROL SHIFT
1129                              <1> K44:                              ; NOT-CTL-SHIFT
1130 00001561 3C37                <1>       cmp    al, 55               ; PRINT SCREEN KEY?
1131 00001563 7528                <1>       jne    short K45           ; NOT PRINT SCREEN
1132 00001565 F6C710              <1>       test   bh, KBX                    ; IS THIS ENHANCED KEYBOARD?
1133 00001568 7407                <1>       jz     short K44A          ; NO, TEST FOR SHIFT STATE
1134 0000156A F6C702              <1>       test   bh, LC_E0           ; YES, LAST CODE A MARKER?
1135 0000156D 7507                <1>       jnz    short K44B          ; YES, IS PRINT SCREEN
1136 0000156F EB41                <1>       jmp    short K45C          ; NO, TRANSLATE TO '*' CHARACTER
1137                              <1> K44A:
1138 00001571 F6C303              <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; NOT 101 KBD, SHIFT KEY DOWN?
1139 00001574 743C                <1>       jz     short K45C          ; NO, TRANSLATE TO '*' CHARACTER
1140                              <1>       ;
1141                              <1>       ;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
1142                              <1> K44B:
1143 00001576 B0AE                <1>       mov    al, ENA_KBD         ; INSURE KEYBOARD IS ENABLED
1144 00001578 E81B010000          <1>       call   SHIP_IT                    ; EXECUTE ENABLE
1145 0000157D B020                <1>       mov    al, EOI                    ; END OF CURRENT INTERRUPT
1146 0000157F E620                <1>       out    20h, al ;out INTA00, al    ; SO FURTHER THINGS CAN HAPPEN
1147                              <1>       ; Print Screen !!!         ; ISSUE PRINT SCREEN INTERRUPT (INT 05h)
1148                              <1>       ;PUSH  BP                  ; SAVE POINTER
1149                              <1>       ;INT   5H                  ; ISSUE PRINT SCREEN INTERRUPT
1150                              <1>       ;POP   BP                  ; RESTORE POINTER
1151 00001581 8025[A86F0000]FC    <1>         and     byte [KB_FLAG_3], ~(LC_E0+LC_E1) ; ZERO OUT THESE FLAGS
1152 00001588 E90CFEFFFF          <1>         jmp     K27                     ; GO BACK WITHOUT EOI OCCURRING
1153                              <1>       ;
1154                              <1>       ;----- HANDLE IN-CORE KEYS
1155                              <1> K45:                              ; NOT-PRINT-SCREEN
1156 0000158D 3C3A                <1>       cmp    al, 58               ; TEST FOR IN-CORE AREA
1157 0000158F 7734                <1>       ja     short K46           ; JUMP IF NOT
1158 00001591 3C35                <1>       cmp    al, 53               ; IS THIS THE '/' KEY?
1159 00001593 7505                <1>       jne    short K45A          ; NO, JUMP
1160 00001595 F6C702              <1>       test   bh, LC_E0           ; WAS THE LAST CODE THE MARKER?
1161 00001598 7518                <1>       jnz    short K45C          ; YES, TRANSLATE TO CHARACTER
1162                              <1> K45A:
1163 0000159A B91A000000          <1>       mov    ecx, 26                  ; LENGHT OF SEARCH
1164 0000159F BF[726E0000]        <1>       mov    edi, K30+10         ; POINT TO TABLE OF A-Z CHARS
1165 000015A4 F2AE                <1>       repne  scasb               ; IS THIS A LETTER KEY?
1166                              <1>              ; 20/02/2015
1167 000015A6 7505                <1>       jne    short K45B              ; NO, SYMBOL KEY
1168                              <1>       ;
1169 000015A8 F6C340              <1>       test   bl, CAPS_STATE             ; ARE WE IN CAPS_LOCK?
1170 000015AB 750C                <1>       jnz    short K45D          ; TEST FOR SURE
1171                              <1> K45B:
1172 000015AD F6C303              <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
1173 000015B0 750C                <1>       jnz    short K45E          ; YES, UPPERCASE
1174                              <1>                                  ; NO, LOWERCASE
1175                              <1> K45C:
1176 000015B2 BB[F46E0000]        <1>       mov    ebx, K10            ; TRANSLATE TO LOWERCASE LETTERS
1177 000015B7 EB51                <1>       jmp    short K56
1178                              <1> K45D:                             ; ALMOST-CAPS-STATE
1179 000015B9 F6C303              <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; CL ON. IS SHIFT ON, TOO?
1180 000015BC 75F4                <1>       jnz    short K45C          ; SHIFTED TEMP OUT OF CAPS STATE
1181                              <1> K45E:
1182 000015BE BB[4C6F0000]        <1>       mov    ebx, K11            ; TRANSLATE TO UPPER CASE LETTERS
1183 000015C3 EB45                <1> K45F: jmp    short K56
1184                              <1>       ;
1185                              <1>       ;----- TEST FOR KEYS F1 - F10
1186                              <1> K46:                              ; NOT IN-CORE AREA
1187 000015C5 3C44                <1>       cmp    al, 68               ; TEST FOR F1 - F10
1188                              <1>       ;ja    short K47           ; JUMP IF NOT
1189                              <1>       ;jmp   short K53           ; YES, GO DO FN KEY PROCESS
1190 000015C7 7635                <1>       jna    short K53
1191                              <1>       ;
1192                              <1>       ;----- HANDLE THE NUMERIC PAD KEYS
1193                              <1> K47:                              ; NOT F1 - F10
1194 000015C9 3C53                <1>       cmp    al, 83               ; TEST NUMPAD KEYS
```

```
1195 000015CB 772D              <1>       ja     short K52          ; JUMP IF NOT
1196                            <1>       ;
1197                            <1>       ;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
1198                            <1> K48:
1199 000015CD 3C4A              <1>       cmp    al , 74                  ; SPECIAL CASE FOR MINUS
1200 000015CF 74ED              <1>       je     short K45E         ; GO TRANSLATE
1201 000015D1 3C4E              <1>       cmp    al , 78                  ; SPECIAL CASE FOR PLUS
1202 000015D3 74E9              <1>       je     short K45E         ; GO TRANSLATE
1203 000015D5 F6C702            <1>       test   bh, LC_E0          ; IS THIS ONE OFTHE NEW KEYS?
1204 000015D8 750A              <1>       jnz    short K49          ; YES, TRANSLATE TO BASE STATE
1205                            <1>       ;
1206 000015DA F6C320            <1>       test   bl, NUM_STATE      ; ARE WE IN NUM LOCK
1207 000015DD 7514              <1>       jnz    short K50          ; TEST FOR SURE
1208 000015DF F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; ARE WE IN SHIFT STATE?
1209                            <1>       ;jnz   short K51          ; IF SHIFTED, REALLY NUM STATE
1210 000015E2 75DA              <1>       jnz    short K45E
1211                            <1>       ;
1212                            <1>       ;----- BASE CASE FOR KEYPAD
1213                            <1> K49:
1214 000015E4 3C4C              <1>       cmp    al, 76             ; SPECIAL CASE FOR BASE STATE 5
1215 000015E6 7504              <1>       jne    short K49A         ; CONTINUE IF NOT KEYPAD 5
1216 000015E8 B0F0              <1>       mov    al, 0F0h           ; SPECIAL ASCII CODE
1217 000015EA EB40              <1>       jmp    short K57          ; BUFFER FILL
1218                            <1> K49A:
1219 000015EC BB[F46E0000]      <1>       mov    ebx, K10           ; BASE CASE TABLE
1220 000015F1 EB27              <1>       jmp    short K64          ; CONVERT TO PSEUDO SCAN
1221                            <1>       ;
1222                            <1>       ;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
1223                            <1> K50:                            ; ALMOST-NUM-STATE
1224 000015F3 F6C303            <1>         test    bl, LEFT_SHIFT+RIGHT_SHIFT
1225 000015F6 75EC              <1>       jnz    short K49          ; SHIFTED TEMP OUT OF NUM STATE
1226 000015F8 EBC4              <1> K51:  jmp    short K45E         ; REALLY NUM STATE
1227                            <1>       ;
1228                            <1>       ;----- TEST FOR THE NEW KEYS ON WT KEYBOARDS
1229                            <1> K52:                            ; NOT A NUMPAD KEY
1230 000015FA 3C56              <1>       cmp    al, 86             ; IS IT THE NEW WT KEY?
1231                            <1>       ;jne   short K53          ; JUMP IF NOT
1232                            <1>       ;jmp   short K45B         ; HANDLE WITH REST OF LETTER KEYS
1233 000015FC 74AF              <1>       je     short K45B
1234                            <1>       ;
1235                            <1>       ;----- MUST BE F11 OR F12
1236                            <1> K53:                            ; F1 - F10 COME HERE, TOO
1237 000015FE F6C303            <1>       test   bl, LEFT_SHIFT+RIGHT_SHIFT ; TEST SHIFT STATE
1238 00001601 74E1              <1>       jz     short K49          ; JUMP, LOWER CASE PSEUDO SC'S
1239                            <1>              ; 20/02/2015
1240 00001603 BB[4C6F0000]      <1>       mov    ebx, K11           ; UPPER CASE PSEUDO SCAN CODES
1241 00001608 EB10              <1>       jmp    short K64          ; TRANSLATE SCAN
1242                            <1>       ;
1243                            <1>       ;----- TRANSLATE THE CHARACTER
1244                            <1> K56:                            ; TRANSLATE-CHAR
1245 0000160A FEC8              <1>       dec    al                 ; CONVERT ORIGIN
1246 0000160C D7                <1>       xlat                      ; CONVERT THE SCAN CODE TO ASCII
1247 0000160D F605[A86F0000]02  <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
1248 00001614 7416              <1>       jz     short K57          ; NO, GO FILL BUFFER
1249 00001616 B4E0              <1>       mov    ah, MC_E0          ; YES, PUT SPECIAL MARKER IN AH
1250 00001618 EB12              <1>       jmp    short K57          ; PUT IT INTO THE BUFFER
1251                            <1>       ;
1252                            <1>       ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
1253                            <1> K64:                            ; TRANSLATE-SCAN-ORGD
1254 0000161A FEC8              <1>       dec    al                 ; CONVERT ORIGIN
1255 0000161C D7                <1>              xlat                       ; CTL TABLE SCAN
1256 0000161D 88C4              <1>       mov    ah, al             ; PUT VALUE INTO AH
1257 0000161F B000              <1>       mov    al, 0              ; ZERO ASCII CODE
1258 00001621 F605[A86F0000]02  <1>       test   byte [KB_FLAG_3], LC_E0   ; IS THIS A NEW KEY?
1259 00001628 7402              <1>       jz     short K57          ; NO, GO FILL BUFFER
1260 0000162A B0E0              <1>       mov    al, MC_E0          ; YES, PUT SPECIAL MARKER IN AL
1261                            <1>       ;
1262                            <1>       ;----- PUT CHARACTER INTO BUFFER
1263                            <1> K57:                            ; BUFFER_FILL
1264 0000162C 3CFF              <1>       cmp    al, -1             ; IS THIS AN IGNORE CHAR
1265                            <1>         ;je  short K59          ; YES, DO NOTHING WITH IT
1266 0000162E 0F8459FDFFFF      <1>       je      K26               ; YES, DO NOTHING WITH IT
1267 00001634 80FCFF            <1>       cmp    ah, -1             ; LOOK FOR -1 PSEUDO SCAN
1268                            <1>         ;jne short K61          ; NEAR_INTERRUPT_RETURN
1269 00001637 0F8450FDFFFF      <1>       je      K26               ; INTERRUPT_RETURN
1270                            <1> ;K59:                           ; NEAR_INTERRUPT_RETURN
1271                            <1> ;     jmp    K26                ; INTERRUPT_RETURN
1272                            <1>
1273                            <1> _K60: ; 29/01/2016
1274 0000163D 80FC68            <1>       cmp    ah, 68h      ; ALT + F1 key
1275 00001640 721F              <1>       jb     short K61
1276 00001642 80FC6F            <1>       cmp    ah, 6Fh ; ALT + F8 key
1277 00001645 771A              <1>       ja     short K61
1278                            <1>       ;
1279 00001647 8A1D[468A0100]    <1>       mov    bl, [ACTIVE_PAGE]
1280 0000164D 80C368            <1>       add    bl, 68h
1281 00001650 38E3              <1>       cmp    bl, ah
1282 00001652 740D              <1>       je     short K61
1283 00001654 6650              <1>       push   ax
1284 00001656 88E0              <1>       mov    al, ah
1285 00001658 2C68              <1>       sub    al, 68h
1286 0000165A E858090000        <1>       call   set_active_page
1287 0000165F 6658              <1>       pop    ax
1288                            <1> K61:                            ; NOT-CAPS-STATE
1289 00001661 8B1D[B66F0000]    <1>       mov    ebx, [BUFFER_TAIL] ; GET THE END POINTER TO THE BUFFER
1290 00001667 89DE              <1>       mov    esi, ebx           ; SAVE THE VALUE
1291 00001669 E857FAFFFF        <1>       call   _K4                ; ADVANCE THE TAIL
1292 0000166E 3B1D[B26F0000]    <1>       cmp    ebx, [BUFFER_HEAD] ; HAS THE BUFFER WRAPPED AROUND
1293 00001674 740E              <1>       je     short K62          ; BUFFER_FULL_BEEP
1294 00001676 668906            <1>       mov    [esi], ax          ; STORE THE VALUE
1295 00001679 891D[B66F0000]    <1>       mov    [BUFFER_TAIL], ebx ; MOVE THE POINTER UP
1296 0000167F E909FDFFFF        <1>       jmp    K26
1297                            <1>       ;;cli                     ; TURN OFF INTERRUPTS
1298                            <1>       ;;mov  al, EOI                 ; END OF INTERRUPT COMMAND
1299                            <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:
00001684 B020           <1>       mov    al, EOI                   ; ENABLE INTERRUPT CONTROLLER CHIP
00001686 E620           <1>       out    INTA00, al
00001688 66B9A602       <1>       mov    cx, 678                   ; DIVISOR FOR 1760 HZ
0000168C B304           <1>       mov    bl, 4              ; SHORT BEEP COUNT (1/16 + 1/64 DELAY)
0000168E E86D0D0000     <1>       call   beep               ; GO TO COMMON BEEP HANDLER
00001693 E901FDFFFF     <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>       ;
00001698 6650           <1>       push   ax                 ; SAVE DATA TO SEND
                        <1>
                        <1>       ;----- WAIT FOR COMMAND TO ACCEPTED
0000169A FA             <1>       cli                       ; DISABLE INTERRUPTS TILL DATA SENT
                        <1>       ; xor  ecx, ecx           ; CLEAR TIMEOUT COUNTER
0000169B B900000100     <1>       mov    ecx, 10000h
                        <1> S10:
000016A0 E464           <1>       in     al, STATUS_PORT           ; READ KEYBOARD CONTROLLER STATUS
000016A2 A802           <1>       test   al, INPT_BUF_FULL  ; CHECK FOR ITS INPUT BUFFER BUSY
000016A4 E0FA           <1>       loopnz S10                ; WAIT FOR COMMAND TO BE ACCEPTED
                        <1>
000016A6 6658           <1>       pop    ax                 ; GET DATA TO SEND
000016A8 E664           <1>       out    STATUS_PORT, al           ; SEND TO KEYBOARD CONTROLLER
000016AA FB             <1>       sti                       ; ENABLE INTERRUPTS AGAIN
000016AB C3             <1>       retn                      ; RETURN TO CALLER
                        <1>
                        <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>       ;
000016AC 6650           <1>       push   ax                 ; SAVE REGISTERS
000016AE 6653           <1>       push   bx
000016B0 51             <1>       push   ecx
000016B1 88C7           <1>       mov    bh, al             ; SAVE TRANSMITTED BYTE FOR RETRIES
000016B3 B303           <1>       mov    bl, 3              ; LOAD RETRY COUNT
                        <1> SD0:
000016B5 FA             <1>       cli                       ; DISABLE INTERRUPTS
000016B6 8025[A76F0000]CF <1>     and    byte [KB_FLAG_2], ~(KB_FE+KB_FA) ; CLEAR ACK AND RESEND FLAGS
                        <1>       ;
                        <1>       ;----- WAIT FOR COMMAND TO BE ACCEPTED
000016BD B900000100     <1>       mov    ecx, 10000h        ; MAXIMUM WAIT COUNT
                        <1> SD5:
000016C2 E464           <1>       in     al, STATUS_PORT           ; READ KEYBOARD PROCESSOR STATUS PORT
000016C4 A802           <1>       test   al, INPT_BUF_FULL  ; CHECK FOR ANY PENDING COMMAND
000016C6 E0FA           <1>       loopnz SD5                ; WAIT FOR COMMAND TO BE ACCEPTED
                        <1>       ;
000016C8 88F8           <1>       mov    al, bh             ; REESTABLISH BYTE TO TRANSMIT
000016CA E660           <1>       out    PORT_A, al         ; SEND BYTE
000016CC FB             <1>       sti                       ; ENABLE INTERRUPTS
                        <1>       ;mov   cx, 01A00h         ; LOAD COUNT FOR 10 ms+
000016CD B9FFFF0000     <1>       mov    ecx, 0FFFFh
                        <1> SD1:
000016D2 F605[A76F0000]30 <1>     test   byte [KB_FLAG_2], KB_FE+KB_FA ; SEE IF EITHER BIT SET
000016D9 750F           <1>       jnz    short SD3          ; IF SET, SOMETHING RECEIVED GO PROCESS
000016DB E2F5           <1>       loop   SD1                ; OTHERWISE WAIT
                        <1> SD2:
000016DD FECB           <1>       dec    bl                 ; DECREMENT RETRY COUNT
000016DF 75D4           <1>       jnz    short SD0          ; RETRY TRANSMISSION
000016E1 800D[A76F0000]80 <1>     or     byte [KB_FLAG_2], KB_ERR ; TURN ON TRANSMIT ERROR FLAG
000016E8 EB09           <1>       jmp    short SD4          ; RETRIES EXHAUSTED FORGET TRANSMISSION
                        <1> SD3:
000016EA F605[A76F0000]10 <1>     test   byte [KB_FLAG_2], KB_FA ; SEE IF THIS IS AN ACKNOWLEDGE
000016F1 74EA           <1>       jz     short SD2          ; IF NOT, GO RESEND
                        <1> SD4:
000016F3 59             <1>       pop    ecx                ; RESTORE REGISTERS
000016F4 665B           <1>       pop    bx
000016F6 6658           <1>       pop    ax
000016F8 C3             <1>       retn                      ; RETURN, GOOD TRANSMISSION
                        <1>
                        <1> SND_LED:
                        <1>       ; -------------------------------------------------------------------------------
                        <1>       ; SND_LED
                        <1>       ;     THIS ROUTINES TURNS ON THE MODE INDICATORS.
                        <1>       ;
                        <1>       ;--------------------------------------------------------------------------------
                        <1>       ;
000016F9 FA             <1>       cli                       ; TURN OFF INTERRUPTS
000016FA F605[A76F0000]40 <1>     test   byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
00001701 755F           <1>       jnz    short SL1          ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
                        <1>       ;
00001703 800D[A76F0000]40 <1>     or     byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
0000170A B020           <1>       mov    al, EOI                   ; END OF INTERRUPT COMMAND
0000170C E620           <1>       out    20h, al ;out INTA00, al   ; SEND COMMAND TO INTERRUPT CONTROL PORT
0000170E EB11           <1>       jmp    short SL0          ; GO SEND MODE INDICATOR COMMAND
                        <1> SND_LED1:
00001710 FA             <1>       cli                       ; TURN OFF INTERRUPTS
00001711 F605[A76F0000]40 <1>     test   byte [KB_FLAG_2], KB_PR_LED ; CHECK FOR MODE INDICATOR UPDATE
```

```
1405 00001718 7548                <1>       jnz    short SL1           ; DON'T UPDATE AGAIN IF UPDATE UNDERWAY
1406                              <1>       ;
1407 0000171A 800D[A76F0000]40    <1>       or     byte [KB_FLAG_2], KB_PR_LED ; TURN ON UPDATE IN PROCESS
1408                              <1> SL0:
1409 00001721 B0ED                <1>       mov    al, LED_CMD         ; LED CMD BYTE
1410 00001723 E884FFFFFF          <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
1411 00001728 FA                  <1>       cli
1412 00001729 E836000000          <1>       call   MAKE_LED            ; GO FORM INDICATOR DATA BYTE
1413 0000172E 8025[A76F0000]F8    <1>       and    byte [KB_FLAG_2], 0F8h    ; ~KB_LEDS ; CLEAR MODE INDICATOR BITS
1414 00001735 0805[A76F0000]      <1>       or     [KB_FLAG_2], al     ; SAVE PRESENT INDICATORS FOR NEXT TIME
1415 0000173B F605[A76F0000]80    <1>       test   byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
1416 00001742 750F                <1>       jnz    short SL2           ; IF SO, BYPASS SECOND BYTE TRANSMISSION
1417                              <1>       ;
1418 00001744 E863FFFFFF          <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
1419 00001749 FA                  <1>       cli                        ; TURN OFF INTERRUPTS
1420 0000174A F605[A76F0000]80    <1>       test   byte [KB_FLAG_2], KB_ERR ; TRANSMIT ERROR DETECTED
1421 00001751 7408                <1>       jz     short SL3           ; IF NOT, DON'T SEND AN ENABLE COMMAND
1422                              <1> SL2:
1423 00001753 B0F4                <1>       mov    al, KB_ENABLE       ; GET KEYBOARD CSA ENABLE COMMAND
1424 00001755 E852FFFFFF          <1>       call   SND_DATA            ; SEND DATA TO KEYBOARD
1425 0000175A FA                  <1>       cli                        ; TURN OFF INTERRUPTS
1426                              <1> SL3:
1427 0000175B 8025[A76F0000]3F    <1>       and    byte [KB_FLAG_2], ~(KB_PR_LED+KB_ERR) ; TURN OFF MODE INDICATOR
1428                              <1> SL1:                              ; UPDATE AND TRANSMIT ERROR FLAG
1429 00001762 FB                  <1>       sti                        ; ENABLE INTERRUPTS
1430 00001763 C3                  <1>       retn                       ; RETURN TO CALLER
1431                              <1>
1432                              <1> MAKE_LED:
1433                              <1>       ;--------------------------------------------------------------------------------
1434                              <1>       ; MAKE_LED
1435                              <1>       ;      THIS ROUTINES FORMS THE DATA BYTE NECESSARY TO TURN ON/OFF
1436                              <1>       ;      THE MODE INDICATORS.
1437                              <1>       ;--------------------------------------------------------------------------------
1438                              <1>       ;
1439                              <1>       ;push  cx                  ; SAVE CX
1440 00001764 A0[A56F0000]        <1>       mov    al, [KB_FLAG]       ; GET CAPS & NUM LOCK INDICATORS
1441 00001769 2470                <1>       and    al, CAPS_STATE+NUM_STATE+SCROLL_STATE ; ISOLATE INDICATORS
1442                              <1>       ;mov   cl, 4               ; SHIFT COUNT
1443                              <1>       ;rol   al, cl              ; SHIFT BITS OVER TO TURN ON INDICATORS
1444 0000176B C0C004              <1>       rol    al, 4 ; 20/02/2015
1445 0000176E 2407                <1>       and    al, 07h                   ; MAKE SURE ONLY MODE BITS ON
1446                              <1>       ;pop   cx
1447 00001770 C3                  <1>       retn                       ; RETURN TO CALLER
1448                              <1>
1449                              <1> ; % include 'kybdata.s'   ; KEYBOARD DATA
1450                              <1>
1451                              <1>
1452                              <1> ; /// End Of KEYBOARD FUNCTIONS ///
2661
2662                                  %include 'video.s' ; 07/03/2015
   1                              <1> ; ****************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - video.s
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 12/02/2021
   5                              <1> ; ----------------------------------------------------------------------------
   6                              <1> ; Beginning: 16/01/2016
   7                              <1> ; ----------------------------------------------------------------------------
   8                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
   9                              <1> ; ----------------------------------------------------------------------------
  10                              <1> ; Turkish Rational DOS
  11                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                              <1> ;
  13                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  14                              <1> ; video.inc (13/08/2015)
  15                              <1> ;
  16                              <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
  17                              <1> ; ****************************************************************************
  18                              <1>
  19                              <1> ; Retro UNIX 386 v1 Kernel - VIDEO.INC
  20                              <1> ; Last Modification: 13/08/2015
  21                              <1> ;             (Video Data is in 'VIDATA.INC')
  22                              <1> ;
  23                              <1> ; //////// VIDEO (CGA) FUNCTIONS ////////////////
  24                              <1>
  25                              <1> ; 16/01/2016 (32 bit modifications, TRDOS386 - TRDOS v2.0, video.s)
  26                              <1> ; INT 31H (TRDOS 386) = INT 10H (IBM PC/AT REAL MODE)
  27                              <1>
  28                              <1> ; IBM PC-AT BIOS Source Code
  29                              <1> ; TITLE VIDEO1 --- 06/10/85 VIDEO DISPLAY BIOS
  30                              <1>
  31                              <1> _int10h:
  32                              <1>       ; 23/03/2016
  33                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
  34 00001771 9C                  <1>       pushfd
  35 00001772 0E                  <1>       push   cs
  36 00001773 E851000000          <1>       call   VIDEO_IO_1
  37 00001778 C3                  <1>       retn
  38                              <1>
  39                              <1> ;--- INT 10 H ----------------------------------------------------------------
  40                              <1> ; VIDEO_IO                                                :
  41                              <1> ;     THESE ROUTINES PROVIDE THE CRT DISPLAY INTERFACE                  :
  42                              <1> ;     THE FOLLOWING FUNCTIONS ARE PROVIDED:                            :
  43                              <1> ;                                                                      :
  44                              <1> ;    (AH)= 00H     SET MODE (AL) CONTAINS MODE VALUE                           :
  45                              <1> ;            (AL) = 00H  40X25 BW MODE (POWER ON DEFAULT)                :
  46                              <1> ;            (AL) = 01H  40X25 COLOR                                     :
  47                              <1> ;            (AL) = 02H  80X25 BW                                        :
  48                              <1> ;            (AL) = 03H  80X25 COLOR                                     :
  49                              <1> ;                          GRAPHICS MODES                                :
  50                              <1> ;            (AL) = 04H  320X200 COLOR                                   :
  51                              <1> ;            (AL) = 05H  320X200 BW MODE                                 :
  52                              <1> ;            (AL) = 06H  640X200 BW MODE                                 :
  53                              <1> ;            (AL) = 07H   80X25 MONOCHROME (USED INTERNAL TO VIDEO ONLY)       :
  54                              <1> ;           *** NOTES -BW MODES OPERATE SAME AS COLOR MODES, BUT COLOR :
  55                              <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>
00001779 [2A1B0000]     <1> M1:   dd    SET_MODE     ; TABLE OF ROUTINES WITHIN VIDEO I/O
0000177D [F71E0000]     <1>       dd    SET_CTYPE
00001781 [2B1F0000]     <1>       dd    SET_CPOS
00001785 [531F0000]     <1>       dd    READ_CURSOR
                        <1>       ;dd   VIDEO_RETURN ; READ_LPEN
00001789 [281B0000]     <1>       dd    set_mode_ncm ; Set mode without clearing video memory
0000178D [991F0000]     <1>       dd    ACT_DISP_PAGE
00001791 [2B200000]     <1>       dd    SCROLL_UP
00001795 [55210000]     <1>       dd    SCROLL_DOWN
00001799 [D5210000]     <1>       dd    READ_AC_CURRENT
0000179D [32220000]     <1>       dd    WRITE_AC_CURRENT
000017A1 [58220000]     <1>       dd    WRITE_C_CURRENT
000017A5 [9D2B0000]     <1>       dd    SET_COLOR
000017A9 [082C0000]     <1>       dd    WRITE_DOT
000017AD [D32B0000]     <1>       dd    READ_DOT
000017B1 [E8220000]     <1>       dd    WRITE_TTY
000017B5 [101B0000]     <1>       dd    VIDEO_STATE
000017B9 [9C360000]     <1>       dd    vga_pal_funcs; 10/08/2016 (TRDOS 386)
000017BD [15310000]     <1>       dd    font_setup   ; 10/07/2016 (TRDOS 386)
000017C1 [5F1B0000]     <1>       dd    VIDEO_RETURN ; RESERVED
000017C5 [63240000]     <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
000017C9 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
000017CA 80FC14         <1>       cmp   ah, M1L/4
000017CD 7205           <1>       jb    short VGA_func
                        <1>
000017CF 80FC4F         <1>       cmp   ah, 4Fh
000017D2 7532           <1>       jne   short M4 ; invalid !
                        <1>
                        <1> VGA_func: ; 26/11/2020
000017D4 06             <1>       push  es ; *
000017D5 1E             <1>       push  ds ; **                   ; SAVE WORK AND PARAMETER REGISTERS
                        <1>
                        <1>       ; 26/11/2020
000017D6 50             <1>       push  eax ; -
                        <1>
000017D7 66B81000       <1>       mov   ax, KDATA           ; POINT DS: TO DATA SEGMENT
000017DB 8ED8           <1>       mov   ds, ax
000017DD 8EC0           <1>       mov   es, ax
                        <1>
                        <1>       ; 26/11/2020
000017DF 58             <1>       pop   eax ; +
                        <1>       ;
000017E0 FB             <1>       sti
000017E1 80FC4F         <1>       cmp   ah, 4Fh
000017E4 747A           <1>       je    short VBE_func
                        <1>
                        <1>       ; 04/12/2020
000017E6 A3[A0960100]   <1>       mov   [video_eax], eax
                        <1>
                        <1>       ; 21/12/2020
000017EB 803D[DA6F0000]FF   <1>       cmp   byte [CRT_MODE], 0FFh     ; Current mode is a VESA VBE mode ?
```

```
000017F2 7213                <1>       jb     short VGA_func_std
                             <1>
000017F4 08E4                <1>       or     ah, ah              ; set mode ?
000017F6 750F                <1>       jnz    short VGA_func_std  ; no
                             <1>
000017F8 803D[5C090000]03    <1>       cmp    byte [vbe3], 3            ; (real) VESA VBE 3 video bios ?
000017FF 7506                <1>       jne    short VGA_func_std  ; no
                             <1>
00001801 E989010000          <1>       jmp    vesa_vbe3_pmi
                             <1>
                             <1>       ; 21/12/2020
                             <1> M4:                              ; COMMAND NOT VALID
00001806 CF                  <1>       iretd                      ; DO NOTHING IF NOT IN VALID RANGE
                             <1>
                             <1> VGA_func_std:
                             <1>       ; 06/12/2020
                             <1>       ; 03/12/2020
00001807 80FC0F              <1>       cmp    ah, 0Fh
0000180A 773D                <1>       ja     short VGA_funcs_0   ; only CGA funcs will be handled by
                             <1>       ; 06/12/2020               ; vga bios firmware
                             <1>       ; 03/12/2020
                             <1>       ;test  ah, 7Fh ; set mode ?
                             <1>       ;;or   ah, ah              ; only 'set mode' will be handled by
                             <1>       ;jnz   short VGA_funcs_0   ; vga bios firmware
                             <1>       ;jz    short vbe_pmi32_0
                             <1>
                             <1>       ; 28/11/2020
0000180C 803D[1C120300]00    <1>       cmp    byte [pmi32], 0           ; 32 bit protected mode interface for
00001813 7634                <1>       jna    short VGA_funcs_0   ; video hardware's vga bios firmware
                             <1>                                  ; ([pmi32] > 0 if it is activated)
                             <1>                                  ; note:
                             <1>                                  ; [vbe3] = 3 is required to activate
                             <1>       ; 07/12/2020
00001815 20E4                <1>       and    ah, ah ; is this set mode ?
00001817 7413                <1>       jz     short vbe_pmi32_2 ; yes
                             <1>
00001819 80FC04              <1>       cmp    ah, 04h       ; set mode ('no clear memory' option)
0000181C 742B                <1>       je     short VGA_funcs_0
                             <1>
                             <1>       ; 07/12/2020
0000181E 803D[DA6F0000]07    <1>       cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
00001825 7622                <1>       jna    short VGA_funcs_0  ; no
                             <1>
                             <1>       ; when mode 3 is active,
                             <1>       ; video bios functions are not redirected
                             <1>       ; to VESA VBE3 PMI except 'set mode' function
                             <1>
                             <1> vbe_pmi32_0:
                             <1>       ; 06/12/2020
                             <1>       ;or    ah, ah
                             <1>       ;jnz   short vbe_pmi32_2
                             <1>       ; ah = 0 ; 'set mode'
                             <1>       ;cmp   al, 3 ; 80x25 text mode, 16 colors, default mode for MainProg
                             <1>       ;jne   short vbe_pmi32_1
                             <1>       ;cmp   byte [CRT_MODE], 3 ; If current video mode <> 3 and requested
                             <1>       ;                         ; video mode is 3, use internal 'set mode';
                             <1>       ;                         ; otherwise, use vesa vbe 3 bios's 'set mode'.
                             <1>       ;jne   short VGA_funcs_0
                             <1> vbe_pmi32_1:
00001827 E963010000          <1>       jmp    vesa_vbe3_pmi
                             <1> ;vbe_pmi32_2:
                             <1>       ;cmp   ah, 04h ; set mode (no clear mem option)
                             <1>       ;jne   short vbe_pmi32_1
                             <1>
                             <1> vbe_pmi32_2:
                             <1>       ; 07/12/2020
0000182C 803D[DA6F0000]07    <1>       cmp    byte [CRT_MODE], 7 ; current mode > 7 ?
00001833 77F2                <1>       ja     short vbe_pmi32_1  ; yes
                             <1>
00001835 3C07                <1>       cmp    al, 7  ; requested mode > 7 ?
00001837 7610                <1>       jna    short VGA_funcs_0  ; no (CGA)
                             <1>
00001839 3C13                <1>       cmp    al, 13h
0000183B 76EA                <1>       jna    short vbe_pmi32_1
                             <1>
0000183D A880                <1>       test   al, 80h
0000183F 7408                <1>       jz     short VGA_funcs_0  ; unknown or special
                             <1>
00001841 3C87                <1>       cmp    al, 87h      ; requested mode > 7 ?
                             <1>                           ; (with no clear mem ops)
00001843 7604                <1>       jna    short VGA_funcs_0  ; no (CGA)
                             <1>
00001845 3C93                <1>       cmp    al, 93h
00001847 76DE                <1>       jna    short vbe_pmi32_1
                             <1>
                             <1>       ; > 13h video modes are unknown or special
                             <1>       ; they must be handled by kernel
                             <1>
                             <1>       ; CGA video modes will be handled by kernel
                             <1>
                             <1> VGA_funcs_0:
00001849 52                  <1>       push   edx ; ***
0000184A 51                  <1>       push   ecx ; ****
0000184B 53                  <1>       push   ebx ; *****
0000184C 56                  <1>       push   esi ; ******
0000184D 57                  <1>       push   edi ; *******
0000184E 55                  <1>       push   ebp ; ********
                             <1>
                             <1>       ;mov   [video_eax], eax ; 12/05/2016
0000184F BF00800B00          <1>       mov    edi, 0B8000h        ; GET offset FOR COLOR CARD
                             <1>
                             <1>       ; 23/03/2016
00001854 C0E402              <1>       shl    ah, 2  ; dword            ; TIMES 2 FOR WORD TABLE LOOKUP
00001857 0FB6F4              <1>       movzx  esi, ah                   ; MOVE OFFSET INTO LOOK UP REGISTER (SI)
                             <1>       ;mov   ah, [CRT_MODE]            ; MOVE CURRENT MODE INTO (AH) REGISTER
```

```
                                <1>
                                <1>       ;;15/01/2017
                                <1>       ; 14/01/2017
                                <1>       ; 02/01/2017
                                <1>       ;;mov  byte [intflg], 31h  ; video interrupt
                                <1>       ;sti ; 26/11/2020
                                <1>       ;
                                <1>
0000185A FFA6[79170000]         <1>       JMP    dword [esi+M1]            ; GO TO SELECTED FUNCTION
                                <1>
                                <1> VBE_func:
                                <1>       ; 26/11/2020
                                <1>       ;sti
00001860 55                     <1>       push   ebp ; *** ; 27/11/2020
00001861 56                     <1>       push   esi ; ****
                                <1>
                                <1>       ; Note:
                                <1>       ; ebx, ecx, edx, edi, ebp registers
                                <1>       ; must be saved by VBE functions and
                                <1>       ; eax register must be set
                                <1>       ; (according to VBE 3 standard)
                                <1>       ; before return from this interrupt
                                <1>       ; (every function must restore and set
                                <1>       ; registers except esp, esi, es, ds)
                                <1>
00001862 803D[5C090000]02       <1>       cmp    byte [vbe3], 2
00001869 7741                   <1>       ja     short VESA_VBE3_PMI_CALL ; VBE3 video bios ('PMID')
                                <1>       ;je    short VBE_func_0  ; Bochs/Qemu/VirtualBox emulator
0000186B 726A                   <1>       jb     short VBE_unknown ; invalid ([vbe3] = 0)
                                <1>
                                <1>       ;jmp   VESA_VBE3_PMI_CALL
                                <1>
                                <1> VBE_func_0:
                                <1>       ; Bochs/Plex86 VGAbios VBE extension
                                <1>       ; (TRDOS 386 v2.0.3 can use VBE graphics modes on emulators)
                                <1>       ; BOCHS/QEMU/VIRTUALBOX
                                <1>
0000186D 8A25[5D090000]         <1>       mov    ah, [vbe2bios]
00001873 80FCC0                 <1>       cmp    ah, 0C0h
00001876 725F                   <1>       jb     short VBE_unknown
00001878 80FCC5                 <1>       cmp    ah, 0C5h
0000187B 775A                   <1>       ja     short VBE_unknown
                                <1>
                                <1>       ; TRDOS 386 is running on BOCHS or QEMU
0000187D B44F                   <1>       mov    ah, 4Fh
                                <1> VBE_func_1:
0000187F 0FB6F0                 <1>       movzx  esi, al ; VESA VBE function number
00001882 66C1E602               <1>       shl    si, 2 ; dword
00001886 6683FE14               <1>       cmp    si, N1L
0000188A 734B                   <1>       jnb    short VBE_unknown
                                <1>       ;sti
                                <1>
0000188C FF96[98180000]         <1>       call   dword [esi+N1] ; call VBE function
                                <1>
                                <1>       ;jmp   short VBE_bios_return
                                <1>
                                <1> VBE_bios_return:
00001892 FA                     <1>       cli
00001893 5E                     <1>       pop    esi ; ****
00001894 5D                     <1>       pop    ebp ; *** ; 27/11/2020
00001895 07                     <1>       pop    es  ; **
00001896 1F                     <1>       pop    ds  ; *
00001897 CF                     <1>       iretd
                                <1>
                                <1>       ; 26/11/2020
                                <1> N1:
00001898 [FB180000]             <1>       dd     vbe_biosfn_return_ctrl_info
0000189C [283A0000]             <1>       dd     vbe_biosfn_return_mode_info
000018A0 [E33A0000]             <1>       dd     vbe_biosfn_set_mode
000018A4 [B33B0000]             <1>       dd     vbe_biosfn_return_current_mode
000018A8 [D23B0000]             <1>       dd     vbe_biosfn_save_restore_state
                                <1>       ;dd    vbe_biosfn_display_window_ctrl
                                <1>       ;dd    vbe_biosfn_set_get_log_scanline
                                <1>       ;dd    vbe_biosfn_set_get_disp_start
                                <1>       ;dd    vbe_biosfn_set_get_dac_pal_frm
                                <1>       ;dd    vbe_biosfn_set_get_palette_data
                                <1>
                                <1> N1L   EQU    $ - N1
                                <1>
                                <1>
                                <1> VESA_VBE3_PMI_CALL: ; VESA VBE video bios (firmware) functions
                                <1>                     ; by using VESA VBE3 Protected Mode Interface
                                <1>
                                <1>       ; 29/11/2020
                                <1>       ; 26/11/2020 - TRDOS 386 v2.0.3 video.s
                                <1>
                                <1>       ; We are here because..
                                <1>       ; 'PMID' has been verified by TRDOS 386 v2.0.3 kernel.
                                <1>       ; (Otherwise bochs/plex86 compatible VBE functions and
                                <1>       ;  modes would be used on BOCHS/QEMU/VIRTUALBOX emulators
                                <1>       ;  or only standard/old VGA graphics modes would be used.)
                                <1>
                                <1>       ; (TRDOS 386 v2.0.3 can use VESA VBE graphics modes if
                                <1>       ;  the video bios is full compatible with VBE3 standard)
                                <1>
                                <1>       ; 29/11/2020
                                <1>
000018AC 0FB6F0                 <1>       movzx  esi, al ; VESA VBE 3 function number
000018AF 66C1E602               <1>       shl    si, 2 ; dword
000018B3 6683FE14               <1>       cmp    si, P1L
000018B7 731E                   <1>       jnb    short VBE_unknown
                                <1>       ;sti
                                <1>
000018B9 57                     <1>       push   edi ; *****
                                <1>
```

```
000018BA FF96[C3180000]      <1>       call   dword [esi+P1] ; call VBE 3 function
                             <1>
000018C0 5F                  <1>       pop    edi ; *****
                             <1>
000018C1 EBCF                <1>       jmp    short VBE_bios_return
                             <1>
                             <1> P1:
000018C3 [DD180000]          <1>       dd     vbe3_pmfn_return_ctrl_info
000018C7 [01190000]          <1>       dd     vbe3_pmfn_return_mode_info
000018CB [3E190000]          <1>       dd     vbe3_pmfn_set_mode
000018CF [39190000]          <1>       dd     vbe3_pmfn_return_current_mode
000018D3 [301A0000]          <1>       dd     vbe3_pmfn_save_restore_state
                             <1>       ;dd    vbe3_pmfn_display_window_ctrl
                             <1>       ;dd    vbe3_pmfn_set_get_log_scanline
                             <1>       ;dd    vbe3_pmfn_set_get_disp_start
                             <1>       ;dd    vbe3_pmfn_set_get_dac_pal_frm
                             <1>       ;dd    vbe3_pmfn_set_get_palette_data
                             <1>       ;dd    vbe3_pmfn_return_pmi ; invalid for TRDOS 386 v2
                             <1>       ;dd    vbe3_pmfn_set_get_pixel_clock
                             <1>
                             <1> P1L   EQU    $ - P1
                             <1>
                             <1> ;     ; 29/11/2020
                             <1> ;     mov    edi, VBE3MODEINFOBLOCK >> 4 ; / 16
                             <1> ;
                             <1> ;     cmp    al, 04h
                             <1> ;     jb     short vbe3_pm_f ; function: 4F00h to 4F03h
                             <1> ;     ja     short vbe3_pmi_f5B ; function: 4F05h to 4F0Bh
                             <1> ;
                             <1> ;     ; check buffer length (must be <= 2048 bytes)
                             <1> ;
                             <1> ;     and    dl, dl ; 0
                             <1> ;     jz     short vbe3_pm_f
                             <1> ;                   ; Return Save/Restore State buffer size
                             <1> ;
                             <1> ;     push   ebx  ; buffer address
                             <1> ;     push   edx  ; function: save (01h) or restore (02h)
                             <1> ;     call
                             <1> ;
                             <1> ;vbe3_pm_f03:
                             <1> ;     cmp    al, 2
                             <1> ;     ja     short vbe3_pm_f ; function 4F03h
                             <1> ;     jb     short vbe3_pm_f1
                             <1> ;
                             <1> ;
                             <1> ;vbe3_pm_f1:
                             <1> ;
                             <1> ;
                             <1> ;vbe3_pmi_f5B:
                             <1> ;     cmp    al, 09h
                             <1> ;     jna    short vbe3_pm_f ; funcs 05h to 09h are usable
                             <1> ;
                             <1> ;     cmp    al, 0Bh ; Get/Set pixel clock, last function
                             <1> ;     jne    short VBE_unknown
                             <1> ;                   ; (do not use 'uncertain' functions
                             <1> ;                   ; because of system-user buff transfers)
                             <1> ;vbe3_pm_f:
                             <1> ;     mov    byte [vbe3_pm_fn], al     ; set
                             <1> ;     ; prepare 16 bit pm segments & registers for pmi call
                             <1> ;     call   VESA_VBE3_PM_FUNCTION
                             <1> ;
                             <1> ;
                             <1>
                             <1>       ; 26/11/2020
                             <1> VBE_unknown:
000018D7 66B80001            <1>       mov    ax, 0100h  ; ah = 1 : Function call failed
                             <1>                         ; al = 0 : Function is not supported
                             <1>
000018DB EBB5                <1>       jmp    short VBE_bios_return
                             <1>
                             <1> vbe3_pmfn_return_ctrl_info:
                             <1>       ; 12/12/2020
                             <1>       ;
                             <1>       ; VBE function 4F00h - Return VBE Controller Information
                             <1>       ;
                             <1>       ; Input:
                             <1>       ;      EDI = Pointer to buffer in which to place
                             <1>       ;           VbeInfoBlock structure
                             <1>       ;
                             <1>       ;      AX = 4F00h
                             <1>       ; Output:
                             <1>       ;      AX = VBE return status
                             <1>       ;           AX = 004Fh -> succeeded
                             <1>       ;           AX <> 004Fh -> failed
                             <1>       ;
                             <1>       ; Modified registers: eax (+ edi for kernel's own call)
                             <1>
                             <1>       ; NOTE: TRDOS 386 v2 (v2.0.3) kernel calls this function
                             <1>       ; during startup while cpu is in real mode
                             <1>       ; (by using int 10h, 4F02h) and saves VbeInfoBlock at
                             <1>       ; VBE3INFOBLOCK address (97E00h for TRDOS 386 v2.0.3).
                             <1>       ;
                             <1>       ; So...
                             <1>       ; This VBE function is adjusted to return/move same info
                             <1>       ; from VBE3INFOBLOCK to user's buffer in EDI.
                             <1>
                             <1>       ; int 31h (int 10h) entrance
                             <1>
000018DD 21FF                <1>       and    edi, edi
000018DF 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:
000018E1 57             <1>       push   edi
000018E2 51             <1>       push   ecx
000018E3 BE007E0900     <1>       mov    esi, VBE3INFOBLOCK
000018E8 B900020000     <1>       mov    ecx, 512
000018ED E855020100     <1>       call   transfer_to_user_buffer
000018F2 59             <1>       pop    ecx
000018F3 5F             <1>       pop    edi
000018F4 720F           <1>       jc     short vbe3_func_fail
                        <1>
000018F6 31C0           <1>       xor    eax, eax
000018F8 B04F           <1>       mov    al, 4Fh ; successful
                        <1> ;vbe_ctrl_info_retn:
000018FA 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>
000018FB 21FF           <1>       and    edi, edi
000018FD 7406           <1>       jz     short vbe3_func_fail ; invalid buffer addr !
000018FF 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>
00001901 09FF           <1>       or     edi, edi
00001903 7506           <1>       jnz    short _vbe3_pmfn_return_mode_info
                        <1>
                        <1> vbe3_func_fail:
00001905 B84F010000     <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
                        <1>                         ; al = 4Fh : Function is supported
0000190A C3             <1>       retn
                        <1>
                        <1>       ; jump from '_vbe_biosfn_return_mode_info'
                        <1> _vbe3_pmfn_return_mode_info:
0000190B 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
0000190C BF00060000     <1>       mov    edi, VBE3MODEINFOBLOCK-VBE3SAVERESTOREBLOCK
                        <1>       ; ES selector base address = VBE3SAVERESTOREBLOCK
00001911 E8FC000000     <1>       call   int10h_32bit_pmi
                        <1>
00001916 5F             <1>       pop    edi
                        <1>
```

```
                        <1>       ;cmp   ax, 004Fh
                        <1>       ;jne   short vbe3_func_retn  ; failed !
                        <1>
00001917 21FF           <1>       and    edi, edi
                        <1>       ;jz    short vbe3_func_success
                        <1>       ; 21/12/2020
00001919 741D           <1>       jz     short vbe3_func_retn
                        <1>
0000191B 6683F84F       <1>       cmp    ax, 004Fh
0000191F 7517           <1>       jne    short vbe3_func_retn  ; failed !
                        <1>
00001921 51             <1>       push   ecx
00001922 BE007C0900     <1>       mov    esi, VBE3MODEINFOBLOCK
00001927 B900010000     <1>       mov    ecx, 256
0000192C E816020100     <1>       call   transfer_to_user_buffer
00001931 59             <1>       pop    ecx
00001932 72D1           <1>       jc     short vbe3_func_fail
                        <1>
00001934 31C0           <1>       xor    eax, eax
00001936 B04F           <1>       mov    al, 4Fh ; successful
                        <1> vbe3_func_success:
                        <1> vbe3_func_retn:
00001938 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
00001939 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
0000193E 57                     <1>       push   edi ; *****
0000193F F6C708                 <1>       test   bh, 8  ; Use user specified CRTC values
00001942 7530                   <1>       jnz    short vbe3_sm_3  ; for refresh rate
                                <1> vbe3_sm_4:
00001944 80E7C1                 <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
                                <1>       ;mov   [vbe_mode_x], bh
                                <1>
00001947 803D[DA6F0000]03       <1>       cmp    byte [CRT_MODE], 3 ; is current mode 03h ?
0000194E 7509                   <1>       jne    short vbe3_sm_1    ; no
                                <1>
                                <1>       ; save mode 03h video pages and cursor positions
00001950 57                     <1>       push   edi ; **!***
00001951 51                     <1>       push   ecx ; ******
                                <1>       ;push  esi
                                <1>
00001952 E8CE040000             <1>       call   save_mode3_multiscreen
                                <1>
                                <1>       ;pop   esi
00001957 59                     <1>       pop    ecx ; ******
00001958 5F                     <1>       pop    edi ; **!***
                                <1> vbe3_sm_1:
                                <1>       ; ax = 4F02h
                                <1>       ; bx = video mode number (vbe2 type)
00001959 E8B4000000             <1>       call   int10h_32bit_pmi
                                <1>                     ; call to far call to VBE3 PMI
                                <1>
0000195E 6683F84F               <1>       cmp    ax, 004Fh ; succeeded ?
00001962 750E                   <1>       jne    short vbe3_sm_2
                                <1>       ; set current mode byte/sign to extended (SVGA) mode
00001964 C605[DA6F0000]FF       <1>       mov    byte [CRT_MODE], 0FFh ; VESA VBE mode
                                <1>       ; set current VBE mode word to bx input
0000196B 66891D[1E120300]       <1>       mov    [video_mode], bx
                                <1> vbe3_sm_2:
                                <1>       ; 22/12/2020
00001972 5F                     <1>       pop    edi ; *****
00001973 C3                     <1>       retn
                                <1>
                                <1> vbe3_sm_3:
                                <1>       ; 22/12/2020
                                <1>       ; copy user's CRTCInfoBlock to the buffer
00001974 51                     <1>       push   ecx
00001975 89FE                   <1>       mov    esi, edi
00001977 BF807D0900             <1>       mov    edi, VBE3CRTCINFOBLOCK
0000197C B940000000             <1>       mov    ecx, 64
00001981 E80B020100             <1>       call   transfer_from_user_buffer
00001986 59                     <1>       pop    ecx
                                <1>       ; set offset (es base addr is VBE3SAVERESTOREBLOCK)
00001987 81EF00760900           <1>       sub    edi, VBE3SAVERESTOREBLOCK
0000198D 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
0000198F 20E4                   <1>       and    ah, ah ; 0 = set mode function
00001991 7402                   <1>       jz     short vbe3_pmi_0
00001993 EB76                   <1>       jmp    vbe3_pmi_9
                                <1>
                                <1> vbe3_pmi_0:
                                <1>       ; 07/12/2020
00001995 88C4                   <1>       mov    ah, al
00001997 80E480                 <1>       and    ah, 80h ; 0 or 80h
0000199A 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
0000199C 8825[AF960100]         <1>       mov    byte [noclearmem], ah ; 0 or 80h
                                <1>       ; check current video mode if it is 03h
000019A2 803D[DA6F0000]03       <1>       cmp    byte [CRT_MODE], 3 ; current mode
000019A9 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
000019AB 56                     <1>       push  esi
000019AC 57                     <1>       push  edi
000019AD 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
000019AE E872040000             <1>       call  save_mode3_multiscreen
                                <1>
000019B3 59                     <1>       pop   ecx
000019B4 5F                     <1>       pop   edi
000019B5 5E                     <1>       pop   esi
                                <1> vbe3_pmi_3:
                                <1>       ; 08/12/2020
                                <1>       ; 07/12/2020
                                <1>       ; case 3 or case 4
000019B6 A2[DA6F0000]           <1>       mov   [CRT_MODE], al
000019BB 3C03                   <1>       cmp   al, 3
000019BD 7407                   <1>       je    short vbe3_pmi_4
                                <1>       ; case 4:
                                <1>       ; Current mode is not 03h and also new mode is not 03h
000019BF 800D[AD960100]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>
000019C6 E847000000             <1>       call  int10h_32bit_pmi
                                <1>
000019CB 803D[DA6F0000]03       <1>       cmp   byte [CRT_MODE], 3  ; new video mode
                                <1>       ;jne  vbe3_pmi_8   ; video mode <> 03h
000019D2 7532                   <1>       jne   short vbe3_pmi_8
                                <1>
                                <1>       ;push  eax ; 04/12/2020
000019D4 53                     <1>       push  ebx
000019D5 51                     <1>       push  ecx
000019D6 52                     <1>       push  edx
000019D7 57                     <1>       push  edi ; 03/12/2020
                                <1>
                                <1>       ; 12/12/2020
000019D8 56                     <1>       push  esi
000019D9 E87A040000             <1>       call  restore_mode3_multiscreen
000019DE 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>       ;
                                <1>       ;; ([p_crt_mode] = 03h)
                                <1>       ;
                                <1>       ;; New video mode is 3 while current video mode is not 3
                                <1>       ;; (multi screen) video pages will be restored from 098000h
                                <1>       ;
                                <1>       ;; restore video pages and cursor positions
                                <1>       ;
                                <1>       ;mov  [ACTIVE_PAGE], al ; current mode 3 active page
                                <1>       ;
                                <1>       ;push  esi
                                <1>       ;
                                <1>       ;; restore video pages
                                <1>       ;mov  esi, 98000h
                                <1>       ;mov  edi, 0B8000h
                                <1>       ;;mov  ecx, 2000h
                                <1>       ;mov  cx, 2000h ; 8K dwords (32K)
                                <1>       ;rep  movsd
                                <1>       ;
                                <1>       ;mov  [p_crt_mode], cl ; reset ('case 3' end condition)
                                <1>       ;
                                <1>       ;; restore cursor positions
                                <1>       ;mov  esi, cursor_pposn
                                <1>       ;mov  edi, CURSOR_POSN
                                <1>       ;;mov  ecx, 4 ; restore all cursor positions (16 bytes)
                                <1>       ;mov  cl, 4
                                <1>       ;rep  movsd
                                <1>       ;
                                <1>       ;pop  esi
                                <1>       ;
```

```
1001                                    <1>       ;; 07/12/2020
1002                                    <1>       ;; restore CRT_START according to ACTIVE_PAGE
1003                                    <1>       ;mov   [CRT_START], cx ; 0
1004                                    <1>       ;
1005                                    <1>       ;; check active page and set it again if it is not 0
1006                                    <1>       ;or    al, al
1007                                    <1>       ;jz    short vbe3_pmi_7
1008                                    <1>       ;
1009                                    <1>       ;mov   cl, al
1010                                    <1> ;vbe3_pmi_5:
1011                                    <1>       ;add   word [CRT_START], 4096
1012                                    <1>       ;dec   cl
1013                                    <1>       ;jnz   short vbe3_pmi_5
1014                                    <1>
1015 000019DF B405                     <1>       mov    ah, 05h ; set current video page
1016                                    <1>       ;al = video page
1017 000019E1 E82C000000               <1>       call   int10h_32bit_pmi
1018                                    <1>
1019                                    <1>       ; check current cursor position & set it again if not 0,0
1020                                    <1>       ;movzx ebx, byte [ACTIVE_PAGE]
1021 000019E6 0FB6D8                   <1>       movzx  ebx, al
1022 000019E9 D0E3                     <1>       shl    bl, 1
1023 000019EB 81C3[368A0100]           <1>       add    ebx, CURSOR_POSN
1024 000019F1 668B13                   <1>       mov    dx, [ebx]
1025 000019F4 6621D2                   <1>       and    dx, dx
1026 000019F7 7409                     <1>       jz     short vbe3_pmi_7
1027                                    <1>
1028                                    <1>       ;dx = cursor position (dl = column, dh = row)
1029                                    <1>       ;mov   bh, [ACTIVE_PAGE] ; 06/12/2020
1030 000019F9 88C7                     <1>       mov    bh, al
1031 000019FB B402                     <1>       mov    ah, 02h ; set cursor position
1032 000019FD E810000000               <1>       call   int10h_32bit_pmi
1033                                    <1>
1034                                    <1>       ;jmp   short vbe3_pmi_7
1035                                    <1>
1036                                    <1> ;vbe3_pmi_6:
1037                                    <1> ;     ; 07/12/2020
1038                                    <1> ;     ; case 1, previous mode is 03h, current mode is 03h
1039                                    <1> ;     ; 03/12/2020
1040                                    <1> ;     cmp    byte [noclearmem], 0
1041                                    <1> ;     jna    short vbe3_pmi_7 ; do not clear memory
1042                                    <1> ;     ; clear video page
1043                                    <1> ;     mov    ecx, 1024 ; 4096/4
1044                                    <1> ;     mov    eax, 07200720h
1045                                    <1> ;     mov    edi, 0B8000h ; [crt_base]
1046                                    <1> ;     add    di, [CRT_START]
1047                                    <1> ;     rep    stosd  ; FILL THE REGEN BUFFER WITH BLANKS
1048                                    <1>
1049                                    <1> vbe3_pmi_7:
1050 00001A02 5F                       <1>       pop    edi
1051 00001A03 5A                       <1>       pop    edx
1052 00001A04 59                       <1>       pop    ecx
1053 00001A05 5B                       <1>       pop    ebx
1054                                    <1>       ;pop   eax ; 04/12/2020
1055                                    <1> vbe3_pmi_8:
1056                                    <1>       ; 04/12/2020
1057                                    <1>       ;(TRDOS 386 v2.0.3, INT 31h, ah=0 return)
1058 00001A06 31C0                     <1>       xor    eax, eax  ; eax = 0 -> succesful
1059                                    <1> vesa_vbe3_pmi_retn:
1060 00001A08 07                       <1>       pop    es  ; **
1061 00001A09 1F                       <1>       pop    ds  ; *
1062 00001A0A CF                       <1>       iretd
1063                                    <1>
1064                                    <1> vbe3_pmi_9:
1065                                    <1>       ; 06/12/2020
1066                                    <1>       ;cmp   ah, 10h ; Set/Get Palette Registers
1067                                    <1>       ;jnb   short vbe3_pmi_10
1068                                    <1>       ; 05/12/2020
1069 00001A0B E802000000               <1>       call   int10h_32bit_pmi
1070 00001A10 EBF6                     <1>       jmp    short vesa_vbe3_pmi_retn
1071                                    <1>
1072                                    <1> ;vbe3_pmi_10:
1073                                    <1>       ; 06/12/2020
1074                                    <1>       ;jmp   VGA_funcs_0
1075                                    <1>
1076                                    <1> int10h_32bit_pmi:
1077                                    <1>       ; 03/12/2020
1078                                    <1>       ; 28/11/2020
1079                                    <1>       ; calling standard VGA Bios (INT 10h) functions
1080                                    <1>       ; by using 32 bit protected mode interface of
1081                                    <1>       ; VESA VBE3 Video Bios (with 'PMID' signature)
1082                                    <1>
1083                                    <1>       ; 03/12/2020
1084                                    <1>       ; eax, ebx, ecx, edx, edi will be used by vbios pmi
1085                                    <1>       ; (esi and ebp will not be used)
1086                                    <1>
1087                                    <1>       ; 03/12/2020
1088 00001A12 56                       <1>       push   esi
1089 00001A13 C1E010                   <1>       shl    eax, 16  ; move function number (ax) to hw
1090 00001A16 8B35[24120300]           <1>       mov    esi, [pmid_addr] ; linear address of
1091                                    <1>                             ; PMInfo.Entrypoint pointer
1092                                    <1>       ;mov   ax, [esi+PMInfo.EntryPoint]
1093 00001A1C 668B06                   <1>       mov    ax, [esi]
1094 00001A1F C1C010                   <1>       rol    eax, 16 ; move PM entry address to hw
1095                                    <1>                      ; and move function number to lw (ax)
1096 00001A22 5E                       <1>       pop    esi
1097                                    <1>
1098                                    <1>       ; top of stack: ; (*)
1099                                    <1>       ; return (the caller) address of "int10h_32bit_pmi"
1100                                    <1>
1101 00001A23 E97BEDFFFF               <1>       jmp    _VBE3PMI_fcall ; will return to the caller (*)
1102                                    <1>
1103                                    <1> _vbe3_pmfn_srs_8:
1104                                    <1>       ; 17/01/2021
1105 00001A28 31DB                     <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
```

```
1106                            <1> _vbe3_pmfn_srs_9: ; 24/01/2021
1107 00001A2A 66B8044F          <1>       mov    ax, 4F04h
1108 00001A2E EBE2              <1>       jmp    short int10h_32bit_pmi
1109                            <1>
1110                            <1> vbe3_pmfn_save_restore_state:
1111                            <1>       ; 24/01/2021
1112                            <1>       ; 23/01/2021
1113                            <1>       ; 16/01/2021, 17/01/2021
1114                            <1>       ; 14/01/2021
1115                            <1>       ;
1116                            <1>       ; VBE function 4F04h - Save/Restore Video State
1117                            <1>       ;
1118                            <1>       ; Input:
1119                            <1>       ;      DL = sub function
1120                            <1>       ;      CL = requested state
1121                            <1>       ;       EBX = pointer to buffer (if DL<>00h)
1122                            <1>       ;      AX = 4F04h
1123                            <1>       ; Output:
1124                            <1>       ;      AX = 004Fh (successful)
1125                            <1>       ;      AH > 0 -> error
1126                            <1>       ;      BX = Number of 64-byte blocks
1127                            <1>       ;           to hold the state buffer (if DL=00h)
1128                            <1>
1129                            <1>       ; Modified registers: eax, ebx, esi, edi
1130                            <1>
1131 00001A30 21DB              <1>       and    ebx, ebx ; user's buffer address
1132 00001A32 750A              <1>       jnz    short _vbe3_pmfn_save_restore_state
1133                            <1>
1134 00001A34 08D2              <1>       or     dl, dl
1135 00001A36 740C              <1>       jz     short _vbe3_pmfn_srs_0
1136                            <1>
1137                            <1>       ; function failed
1138                            <1>       ;;mov  eax, 0100h
1139                            <1>       ;sub   eax, eax
1140                            <1>       ;inc   ah  ; eax = 0100h
1141                            <1>       ;retn
1142                            <1>       ; 16/01/2021
1143                            <1> _vbe3_pmfn_srs_fail:
1144 00001A38 B84F010000        <1>       mov    eax, 014Fh ; ah = 1 : Function call failed
1145                            <1>                         ; al = 4Fh : Function is supported
1146                            <1> _vbe3_srs_retn:
1147 00001A3D C3                <1>       retn
1148                            <1>
1149                            <1> _vbe3_pmfn_save_restore_state:
1150 00001A3E 20D2              <1>       and    dl, dl
1151 00001A40 7559              <1>       jnz    short _vbe3_pmfn_srs_2
1152                            <1> _vbe3_pmfn_srs:
1153 00001A42 31DB              <1>       xor    ebx, ebx
1154                            <1> _vbe3_pmfn_srs_0:
1155                            <1>       ; 24/01/2021
1156 00001A44 83F90F            <1>       cmp    ecx, 0Fh
1157                            <1>       ;ja    short _vbe3_pmfn_srs_1
1158 00001A47 77EF              <1>       ja     short _vbe3_pmfn_srs_fail
1159                            <1>
1160                            <1>       ; !!! CLEAR CL BIT 2 !!!
1161                            <1>       ; (when bit 2 is set, function causes cpu exception)
1162                            <1>       ; BIOS data will not be saved and restored
1163                            <1>       ; (to prevent protected mode page fault error)
1164 00001A49 80E1FD              <1>       and    cl, ~2 ; and cl, not 2
1165                            <1>
1166                            <1>       ; 24/01/2021
1167                            <1>       ;mov   bl, 1
1168 00001A4C FEC3              <1>       inc    bl ; = 1
1169 00001A4E 66D3E3            <1>       shl    bx, cl
1170 00001A51 66231D[28120300]  <1>       and    bx, [vbe3stbsflags]
1171 00001A58 7416              <1>       jz     short _vbe3_pmfn_srs_1
1172                            <1>       ;mov   bx, cx
1173 00001A5A 89CB              <1>       mov    ebx, ecx ; <= 15
1174 00001A5C D0E3              <1>       shl    bl, 1 ; 0, 2, 8 .. 30
1175 00001A5E 668B9B[833C0000]  <1>       mov    bx, [vbestatebufsize+ebx]
1176 00001A65 89DF              <1>       mov    edi, ebx
1177                            <1>       ; edi = state buffer size in bytes
1178 00001A67 66C1EB06          <1>       shr    bx, 6 ; / 64
1179 00001A6B 66B84F00          <1>       mov    ax, 4Fh
1180 00001A6F C3                <1>       retn
1181                            <1> _vbe3_pmfn_srs_1:
1182                            <1>       ; ax = 4F04h
1183                            <1>       ;call  int10h_32bit_pmi
1184                            <1>       ; 24/01/2021
1185                            <1>       ;call  _vbe3_pmfn_srs_8
1186                            <1>       ; ebx = 0
1187 00001A70 E8B5FFFFFF        <1>       call   _vbe3_pmfn_srs_9
1188 00001A75 6683F84F          <1>       cmp    ax, 004Fh
1189 00001A79 75C2              <1>       jne    short _vbe3_srs_retn
1190                            <1>       ; 24/01/2021
1191                            <1>       ;cmp   ecx, 0Fh
1192                            <1>       ;ja    short _vbe3_srs_retn
1193                            <1>       ; 24/01/2021
1194                            <1>       ;mov   ax, 1
1195 00001A7B B001              <1>       mov    al, 1
1196 00001A7D 66D3E0            <1>       shl    ax, cl
1197 00001A80 660905[28120300]  <1>       or     [vbe3stbsflags], ax ; set flag for state option
1198                            <1>       ; 23/01/2021
1199 00001A87 89DF              <1>       mov    edi, ebx
1200 00001A89 89C8              <1>       mov    eax, ecx
1201 00001A8B D0E0              <1>       shl    al, 1
1202 00001A8D 66C1E706          <1>       shl    di, 6 ; * 64
1203 00001A91 6689B8[833C0000]  <1>       mov    [vbestatebufsize+eax], di
1204                            <1>                           ; save buf size for option
1205                            <1>       ;xchg  edi, ebx
1206                            <1>       ; edi = state buffer size in bytes
1207 00001A98 B04F              <1>       mov    al, 4Fh
1208 00001A9A C3                <1>       retn
1209                            <1>
1210                            <1> _vbe3_pmfn_srs_2:
```

```
1211                                    <1>      ; 24/01/2021
1212                                    <1>      ; !!! CLEAR CL BIT 2 !!!
1213                                    <1>      ; (when bit 2 is set, function causes cpu exception)
1214                                    <1>      ; BIOS data will not be saved and restored
1215                                    <1>      ; (to prevent protected mode page fault error)
1216                                    <1>
1217 00001A9B F6C1FD                    <1>      test   cl, ~2 ; test cl, not 2
1218 00001A9E 7498                      <1>      jz     short _vbe3_pmfn_srs_fail
1219                                    <1>
1220 00001AA0 80FA02                    <1>      cmp    dl, 2
1221 00001AA3 7748                      <1>      ja     short _vbe3_pmfn_srs_5
1222                                    <1>
1223                                    <1>      ;and   cl, ~2 ; and cl, not 2
1224                                    <1>
1225 00001AA5 53                        <1>      push   ebx ; * ; buffer address
1226                                    <1>      ; save or restore state
1227                                    <1>      ; (get required buffer size at first)
1228 00001AA6 52                        <1>      push   edx ; **
1229 00001AA7 28D2                      <1>      sub    dl, dl ; 0
1230 00001AA9 E894FFFFFF                <1>      call   _vbe3_pmfn_srs
1231 00001AAE 5A                        <1>      pop    edx ; **
1232                                    <1>      ; 24/01/2021
1233 00001AAF 5B                        <1>      pop    ebx ; *
1234 00001AB0 08E4                      <1>      or     ah, ah
1235 00001AB2 7538                      <1>      jnz    short _vbe3_pmfn_srs_4 ; error
1236                                    <1>
1237                                    <1>      ; edi = buffer size in bytes
1238 00001AB4 81FF00080000              <1>      cmp    edi, 2048
1239 00001ABA 772B                      <1>      ja     short _vbe3_pmfn_srs_3
1240                                    <1>
1241 00001ABC 80FA01                    <1>      cmp    dl, 1
1242 00001ABF 7531                      <1>      jne    short _vbe3_pmfn_srs_6 ; restore state
1243                                    <1>
1244                                    <1>      ; save video state
1245                                    <1>      ;xor   ebx, ebx ; points to VBE3SAVERESTOREBLOCK
1246                                    <1>      ;mov   ax, 4F04h
1247                                    <1>      ;call  int10h_32bit_pmi
1248                                    <1>
1249                                    <1>      ; 24/01/2021
1250 00001AC1 E842000000                <1>      call   _vbe3_pmfn_srs_7
1251                                    <1>
1252 00001AC6 6683F84F                  <1>      cmp    ax, 004Fh
1253 00001ACA 7520                      <1>      jne    short _vbe3_pmfn_srs_4
1254                                    <1>
1255 00001ACC 09DB                      <1>      or     ebx, ebx  ; kernel ('sysvideo') ?
1256 00001ACE 741C                      <1>      jz     short _vbe3_pmfn_srs_4 ; yes
1257                                    <1>
1258                                    <1>      ; the caller is user
1259 00001AD0 51                        <1>      push   ecx ; *
1260 00001AD1 89F9                      <1>      mov    ecx, edi ; state buffer size
1261 00001AD3 BE00760900                <1>      mov    esi, VBE3SAVERESTOREBLOCK ; source
1262                                    <1>                                ; (vbe3 pmi buff)
1263 00001AD8 89DF                      <1>      mov    edi, ebx     ; destination (user buff)
1264 00001ADA E868000100                <1>      call   transfer_to_user_buffer
1265 00001ADF 59                        <1>      pop    ecx ; *
1266 00001AE0 7205                      <1>      jc     short _vbe3_pmfn_srs_3
1267                                    <1>
1268 00001AE2 29C0                      <1>      sub    eax, eax
1269 00001AE4 B04F                      <1>      mov    al, 4Fh
1270 00001AE6 C3                        <1>      retn
1271                                    <1>
1272                                    <1>      ; 24/01/2021
1273                                    <1> _vbe3_pmfn_srs_3:
1274 00001AE7 B84F010000                <1>      mov    eax, 014Fh
1275                                    <1> _vbe3_pmfn_srs_4:
1276 00001AEC C3                        <1>      retn
1277                                    <1> _vbe3_pmfn_srs_5:
1278 00001AED 31C0                      <1>      xor    eax, eax
1279 00001AEF FEC4                      <1>      inc    ah
1280                                    <1>      ; eax = 0100h, function is not supported
1281 00001AF1 C3                        <1>      retn
1282                                    <1>
1283                                    <1> _vbe3_pmfn_srs_6:
1284                                    <1>      ; restore video state
1285                                    <1>      ; 24/01/2021
1286                                    <1>      ;pop   ebx ; *
1287                                    <1>      ; 23/01/2021
1288 00001AF2 09DB                      <1>      or     ebx, ebx ; 0 ?
1289 00001AF4 7412                      <1>      jz     short _vbe3_pmfn_srs_7 ; 'sysvideo' call
1290                                    <1>      ; 24/01/2021
1291                                    <1>      ;jz    _vbe3_pmfn_srs_8
1292 00001AF6 89DE                      <1>      mov    esi, ebx
1293                                    <1>      ; esi = user's video state buffer
1294 00001AF8 51                        <1>      push   ecx ; *
1295 00001AF9 89F9                      <1>      mov    ecx, edi ; state buffer size
1296 00001AFB BF00760900                <1>      mov    edi, VBE3SAVERESTOREBLOCK ; destination
1297                                    <1>                                ; (vbe3 pmi buff)
1298                                    <1>      ;mov   esi, ebx     ; source (user buff)
1299 00001B00 E88C000100                <1>      call   transfer_from_user_buffer
1300 00001B05 59                        <1>      pop    ecx ; *
1301 00001B06 72DF                      <1>      jc     short _vbe3_pmfn_srs_3
1302                                    <1> _vbe3_pmfn_srs_7:
1303 00001B08 53                        <1>      push   ebx ; *
1304                                    <1>      ; restore video state
1305                                    <1>      ;xor   ebx, ebx ; points to VBE3SAVERESTOREBLOCK
1306                                    <1>      ;mov   ax, 4F04h
1307                                    <1>      ;call  int10h_32bit_pmi
1308                                    <1>      ; 17/01/2021
1309 00001B09 E81AFFFFFF                <1>      call   _vbe3_pmfn_srs_8
1310 00001B0E 5B                        <1>      pop    ebx ; *
1311 00001B0F C3                        <1>      retn
1312                                    <1>
1313                                    <1> VIDEO_STATE:
1314                                    <1>      ; 26/06/2016
1315                                    <1>      ; 12/05/2016
```

```
1316                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
1317                                    <1>
1318                                    <1> ;--------------------------------------------------
1319                                    <1> ; VIDEO STATE
1320                                    <1> ;  RETURNS THE CURRENT VIDEO STATE IN AX
1321                                    <1> ;  AH = NUMBER OF COLUMNS ON THE SCREEN
1322                                    <1> ;  AL = CURRENT VIDEO MODE
1323                                    <1> ;  BH = CURRENT ACTIVE PAGE
1324                                    <1> ;--------------------------------------------------
1325                                    <1>
1326 00001B10 8A25[DC6F0000]           <1>       mov    ah, [CRT_COLS]      ; GET NUMBER OF COLUMNS
1327 00001B16 A0[DA6F0000]             <1>       mov    al, [CRT_MODE]      ; CURRENT MODE
1328                                    <1>       ;movzx esi, al
1329                                    <1>       ;mov   ah, [esi+M6]
1330                                    <1>       ; BH = active page
1331 00001B1B 8A3D[468A0100]           <1>       mov    bh, [ACTIVE_PAGE] ; GET CURRENT ACTIVE PAGE
1332 00001B21 FA                       <1>       cli    ; 02/01/2017
1333 00001B22 5D                       <1>       pop    ebp          ; RECOVER REGISTERS
1334 00001B23 5F                       <1>       pop    edi
1335 00001B24 5E                       <1>       pop    esi
1336 00001B25 59                       <1>       pop    ecx          ; DISCARD SAVED BX
1337 00001B26 EB41                     <1>       jmp    short M15    ; RETURN TO CALLER
1338                                    <1>
1339                                    <1> set_mode_ncm:
1340                                    <1>       ; 17/11/2020 (TRDOS 386 v2.0.3)
1341                                    <1>       ; 04/07/2016 - TRDOS 386 (TRDOS v2.0)
1342                                    <1>       ; set mode without clearing the video memory
1343                                    <1>       ; (ony for graphics modes)
1344                                    <1>
1345                                    <1>       ;cmp   al, 7 ; IBM PC CGA modes
1346                                    <1>       ;jna   short SET_MODE ; normal function (clear)
1347                                    <1>       ;; do not clear memory
1348                                    <1>       ;;mov  [noclearmem], al ; > 0
1349                                    <1>       ;mov   byte [noclearmem], 80h ; 17/11/2020
1350                                    <1>       ;call  _set_mode
1351                                    <1>       ;mov   byte [noclearmem], 0
1352                                    <1>         ;jmp     short VIDEO_RETURN
1353                                    <1>
1354                                    <1>       ; 17/11/2020 (TRDOS v2.0.3)
1355 00001B28 0C80                     <1>       or     al, 80h ; not clear memory option
1356                                    <1>
1357                                    <1>       ; 05/12/2020
1358                                    <1>       ; 27/11/2020
1359                                    <1>       ; 17/11/2020
1360                                    <1>       ; 08/08/2016, 10/08/2016
1361                                    <1>       ; 29/07/2016, 30/07/2016
1362                                    <1>       ; 25/07/2016, 26/07/2016, 27/07/2016
1363                                    <1>       ; 02/07/2016, 18/07/2016, 23/07/2016
1364                                    <1>       ; 24/06/2016, 26/06/2016
1365                                    <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
1366                                    <1> SET_MODE:
1367                                    <1>       ; For 32 bit TRDOS and Retro UNIX 386:
1368                                    <1>       ;      valid video mode: 03h only!
1369                                    <1>       ;      (VGA modes will be selected with another routine)
1370                                    <1>       ;
1371                                    <1>       ; set_txt_mode ; 80*25 (16 fore colors, 8 back colors)
1372                                    <1>
1373                                    <1>       ; 27/11/2020
1374                                    <1>
1375                                    <1>       ; Check if current mode is
1376                                    <1>       ; Bochs/Plex86 VBE graphics mode
1377 00001B2A 803D[DA6F0000]FF          <1>       cmp    byte [CRT_MODE], 0FFh ; VESA VBE graphics mode
1378 00001B31 7220                     <1>       jb     short _set_mode_      ; signature
1379                                    <1>                                  ; VBE mode number is in
1380                                    <1>                                  ; [video_mode] bit 0to8
1381 00001B33 88C3                     <1>       mov    bl, al ; save video mode
1382 00001B35 E89D240000               <1>       call   dispi_get_enable
1383 00001B3A 50                       <1>       push   eax ; save current VBE dispi status
1384                                    <1>       ; Disable Bochs/Plex86 VBE dispi
1385                                    <1>       ;mov   ax, 0 ; VBE_DISPI_DISABLED
1386 00001B3B 31C0                     <1>       xor    eax, eax ; 0
1387 00001B3D E869240000               <1>       call   dispi_set_enable
1388 00001B42 88D8                     <1>       mov    al, bl ; restore video mode
1389 00001B44 E827000000               <1>       call   _set_mode
1390 00001B49 58                       <1>       pop    eax ; restore current VBE dispi status
1391 00001B4A 7313                     <1>       jnc    short VIDEO_RETURN
1392                                    <1>       ; ! unimplemented or invalid video mode number !
1393                                    <1>       ; VBE dispi must be enabled again
1394                                    <1>       ; (return to run on current VBE graphics mode)
1395                                    <1>       ;;mov  al, [video_mode+1] ; bit 8 to 15
1396                                    <1>       ;;and  al, 0C0h ; isolate bit 14 and bit 15
1397                                    <1>       ;;or   al, 1 ; VBE_DISPI_ENABLED
1398 00001B4C E85A240000               <1>       call   dispi_set_enable
1399 00001B51 EB07                     <1>       jmp    short _video_func_err
1400                                    <1>
1401                                    <1> _set_mode_:
1402                                    <1>       ; VGA bios (non-VBE) 'setmode' procedure
1403                                    <1>
1404                                    <1>       ; 26/11/2020 (TRDOS v2.0.3)
1405                                    <1>
1406                                    <1> ;--------------------------------------------------
1407                                    <1> ; SET MODE                                       :
1408                                    <1> ;     THIS ROUTINE INITIALIZES THE ATTACHMENT TO    :
1409                                    <1> ;     THE SELECTED MODE, THE SCREEN IS BLANKED.     :
1410                                    <1> ; INPUT                                           :
1411                                    <1> ;     (AL) - MODE SELECTED (RANGE 0-7)       :
1412                                    <1> ; OUTPUT                                         :
1413                                    <1> ;     NONE                                       :
1414                                    <1> ;--------------------------------------------------
1415                                    <1>
1416 00001B53 E818000000               <1>       call   _set_mode ; 24/06/2016 (set_txt_mode)
1417                                    <1>       ; 26/11/2020
1418 00001B58 7305                     <1>       jnc    short VIDEO_RETURN
1419                                    <1>
1420                                    <1>       ; 26/11/2020
```

```
1421                            <1> _video_func_err:
1422 00001B5A 31C0             <1>         xor    eax, eax ; function call failed
1423 00001B5C 48               <1>         dec    eax  ; 0FFFFFFFFh ; - 1
1424 00001B5D EB05             <1>         jmp    short _video_return
1425                            <1>
1426                            <1> ; 12/05/2016
1427                            <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
1428                            <1>
1429                            <1> ;-----      NORMAL RETURN FROM ALL VIDEO RETURNS
1430                            <1>
1431                            <1> VIDEO_RETURN:
1432 00001B5F A1[A0960100]     <1>         mov    eax, [video_eax] ; 12/05/2016
1433                            <1> _video_return:
1434 00001B64 FA               <1>         cli ; 02/01/2017
1435 00001B65 5D               <1>         pop    ebp ; ******** ; 26/11/2020
1436 00001B66 5F               <1>         pop    edi ; *******
1437 00001B67 5E               <1>         pop    esi ; ******
1438 00001B68 5B               <1>         pop    ebx ; *****
1439                            <1> M15:  ; VIDEO_RETURN_C
1440                            <1>         ;;15/01/2017
1441                            <1>         ; 02/01/2017
1442                            <1>         ;;mov  byte [intflg], 0
1443                            <1>         ;
1444 00001B69 59               <1>         pop    ecx ; **** ; 26/11/2020
1445 00001B6A 5A               <1>         pop    edx ; ***
1446 00001B6B 1F               <1>         pop    ds  ; **
1447 00001B6C 07               <1>         pop    es  ; *      ; RECOVER SEGMENTS
1448 00001B6D CF               <1>         iretd        ; ALL DONE
1449                            <1>
1450                            <1> set_txt_mode:
1451                            <1>
1452                            <1>         ; 29/07/2016
1453                            <1>         ; 27/06/2016
1454 00001B6E B003             <1>         mov    al, 3 ; 26/11/2020 (bit 7 = 0)
1455                            <1>
1456                            <1>         ; 17/11/2020 (TRDOS v2.0.3)
1457                            <1>         ;mov  byte [noclearmem], 0
1458                            <1>
1459                            <1> ; 10/08/2016
1460                            <1> ; 08/08/2016
1461                            <1> ; 30/07/2016
1462                            <1> ; 29/07/2016
1463                            <1> ; 25/07/2016, 26/07/2016, 27/07/2016
1464                            <1> ; 07/07/2016, 18/07/2016, 23/07/2016
1465                            <1> ; 02/07/2016, 03/07/2016, 04/07/2016
1466                            <1> ; 26/06/2016
1467                            <1> ; 24/06/2016 (set_txt_mode -> _set_mode)
1468                            <1> ; 17/06/2016
1469                            <1> ; 29/05/2016
1470                            <1> ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
1471                            <1>
1472                            <1> _set_mode:
1473                            <1>         ; 12/12/2020
1474                            <1>         ; 26/11/2020
1475                            <1>         ; call from 'biosfn_set_video_mode'
1476                            <1>         ;      (bochs/plex86 video bios code)
1477                            <1>         ; call from 'SET_MODE'
1478                            <1>         ;      (TRDOS 386 v2 default, IBM PC/AT rom bios code)
1479                            <1>         ; continue from 'set_txt_mode'
1480                            <1>
1481                            <1>         ; INPUT:
1482                            <1>         ;      al = VGA video mode
1483                            <1>         ; RETURN:
1484                            <1>         ;      cf = 1 -> video mode not implemented
1485                            <1>         ;      cf = 0 -> OK
1486                            <1>         ;
1487                            <1>         ; Modified registers: eax, bx, ecx, esi, edi, (ebp)
1488                            <1>
1489                            <1>         ; 17/11/2020 (TRDOS v2.0.3)
1490                            <1>         ; no clear memory option
1491                            <1>         ; (from mode number byte bit 7)
1492 00001B70 88C4             <1>         mov    ah, al
1493 00001B72 80E480           <1>         and    ah, 80h
1494                            <1>         ;mov   [noclearmem], al
1495 00001B75 8825[AF960100]   <1>         mov    [noclearmem], ah
1496                            <1>         ;and   al, 7Fh ; clear bit 7
1497                            <1>         ;;xor  [noclearmem], al ; clear bit 0 to 6
1498                            <1>         ; 26/11/2020
1499 00001B7B 30E0             <1>         xor    al, ah ; and al, 7Fh
1500                            <1>
1501                            <1>         ; 19/11/2020
1502                            <1>
1503                            <1>         ; Video mode 03h action principle:
1504                            <1>         ;
1505                            <1>         ; for case 1:
1506                            <1>         ; Current mode is 03h and next/requested mode is not 03h
1507                            <1>         ;      - save mode (set mode 03h flag)
1508                            <1>         ;      - save 8 video pages (which are will be restored)
1509                            <1>         ;      - save active page number (which will be reactivated)
1510                            <1>         ;      - set active page to 0 always (no multi screen)
1511                            <1>         ;      - save 8 cursor positions (which will be restored)
1512                            <1>         ;      - use 'noclearmem' option
1513                            <1>         ;      [p_crt_mode] = 0 -> 03h
1514                            <1>         ;
1515                            <1>         ; for case 2:
1516                            <1>         ; Current mode is 03h and next/requested mode is also 03h
1517                            <1>         ;      - clear active video page if 'noclearmem' is not set
1518                            <1>         ;      [p_crt_mode] = 0 -> 80h -> 0
1519                            <1>         ;
1520                            <1>         ; for case 3:
1521                            <1>         ; Current mode is not 03h and next/requested mode is 03h
1522                            <1>         ;      - restore video pages (8 video pages were saved)
1523                            <1>         ;      - restore active page number (which were saved)
1524                            <1>         ;      - restore 8 cursor positions (which were saved)
1525                            <1>         ;      - reset/clear mode 03h flag
```

```
1526                                    <1>     ;       [p_crt_mode] = 03h -> 0
1527                                    <1>     ;
1528                                    <1>     ; for case 4:
1529                                    <1>     ; Current mode is not 03h and next/requested mode is not 03h
1530                                    <1>     ;       - use 'noclearmem' option
1531                                    <1>     ;       - set active page to 0 always
1532                                    <1>     ;       [p_crt_mode] = 03h -> 83h -> 03h
1533                                    <1>     ;
1534                                    <1>     ; initial (boot time) values:
1535                                    <1>     ;       [p_crt_mode] = 0 ("there isn't a page backup, yet")
1536                                    <1>     ;         [CRT_MODE] = 3 (kernel's starting mode)
1537                                    <1>
1538                                    <1>     ; 26/11/2020
1539 00001B7D 3C03                    <1>     cmp    al, 03h          ; mode 3, 80x25 text, 16 colors
1540 00001B7F 7515                    <1>     jne    short _sm_0 ; (default mode for TRDOS 386 mainprog)
1541                                    <1>
1542                                    <1>     ; case 2 or case 3
1543                                    <1>
1544                                    <1>     ; check current video mode if it is 03h
1545 00001B81 08E4                    <1>     or     ah, ah ; 80h or 0 ('noclearmem' option)
1546 00001B83 7521                    <1>     jnz    short _sm_1 ; do not clear display page
1547                                    <1>
1548                                    <1>     ; 26/11/2020
1549                                    <1>     ; Note:
1550                                    <1>     ; [CRT_MODE] = 0FFh for VESA VBE video modes
1551                                    <1>     ; [video_mode] = standard VGA and VESA VBE video modes
1552                                    <1>
1553 00001B85 3805[DA6F0000]            <1>     cmp    [CRT_MODE], al      ; 03h
1554 00001B8B 7520                    <1>     jne    short _sm_2  ; case 3 ([p_crt_mode] = 03h)
1555                                    <1>
1556                                    <1>     ; case 2
1557                                    <1>
1558                                    <1>     ; [p_crt_mode] = 0
1559                                    <1>
1560                                    <1>     ; 19/11/2020
1561                                    <1>     ; If '_set_mode' procedure is called for video mode 3
1562                                    <1>     ;     while video mode is 3, video page will be cleared
1563                                    <1>     ;     and cursor position of video page will be reset.
1564                                    <1>
1565                                    <1>     ; clear display page
1566 00001B8D C605[AD960100]80            <1>     mov    byte [p_crt_mode], 80h ; clear page sign
1567 00001B94 EB1C                    <1>     jmp    short _sm_3  ; bypass save video page routine
1568                                    <1> _sm_0:
1569                                    <1>     ; case 1 or case 4
1570                                    <1>
1571                                    <1>     ; 05/12/2020
1572 00001B96 803D[DA6F0000]03        <1>     cmp    byte [CRT_MODE], 3 ; is current mode 03h?
1573 00001B9D 7507                    <1>     jne    short _sm_1  ; case 4 ; [p_crt_mode] = 03h
1574                                    <1>
1575                                    <1>     ; case 1
1576                                    <1>     ; [p_crt_mode] = 0
1577                                    <1>
1578                                    <1>     ; 19/11/2020
1579                                    <1>     ; If '_set_mode' procedure is called for a video mode
1580                                    <1>     ;     except video mode 3 while current video mode
1581                                    <1>     ;     is 3, all video pages of mode 3 will be copied
1582                                    <1>     ;     to 98000h address as backup, before mode change.
1583                                    <1>
1584                                    <1> _sm_save_pm:
1585                                    <1>     ; 12/12/2020
1586                                    <1>     ;; 03/07/2016
1587                                    <1>     ;; save video pages
1588                                    <1>     ;mov   esi, 0B8000h
1589                                    <1>     ;mov   edi, 98000h ; 30/07/2016
1590                                    <1>     ;mov   ecx, (0B8000h-0B0000h)/4
1591                                    <1>     ;rep   movsd
1592                                    <1>
1593                                    <1>     ;mov   byte [p_crt_mode], 3 ; previous mode, backup sign
1594                                    <1>     ;; 26/11/2020
1595                                    <1>     ;xchg  cl, [ACTIVE_PAGE]
1596                                    <1>     ;mov   [p_crt_page], cl  ; save as previous active page
1597                                    <1>     ;
1598                                    <1>     ;; save cursor positions
1599                                    <1>     ;mov   esi, CURSOR_POSN
1600                                    <1>     ;mov   edi, cursor_pposn ; cursor positions backup
1601                                    <1>     ;mov   cl, 4
1602                                    <1>     ;rep   movsd
1603                                    <1>
1604                                    <1>     ; 12/12/2020
1605 00001B9F E881020000              <1>     call   save_mode3_multiscreen
1606                                    <1>
1607                                    <1>     ; 29/07/2016
1608                                    <1>     ;;mov  [ACTIVE_PAGE], cl ; 0
1609                                    <1>     ;xchg  cl, [ACTIVE_PAGE]
1610                                    <1>     ;mov   [p_crt_page], cl  ; previous page (for mode 3)
1611                                    <1>
1612                                    <1>     ; [ACTIVE_PAGE] = 0
1613                                    <1>
1614 00001BA4 EB07                    <1>     jmp    short _sm_2  ; case 1 - 19/11/2020
1615                                    <1> _sm_1:
1616                                    <1>     ; 26/11/2020
1617                                    <1>     ; 19/11/2020
1618                                    <1>
1619                                    <1>     ; case 4
1620 00001BA6 800D[AD960100]80        <1>     or     byte [p_crt_mode], 80h
1621                                    <1>                          ; here [p_crt_mode] must be 83h
1622                                    <1>                          ;        (for case 4)
1623                                    <1>                          ; (because video mode 03h
1624                                    <1>                          ; was changed before as in case 1)
1625                                    <1>
1626                                    <1> _sm_2:      ; case 4 (jump to _sm_2) - 19/11/2020
1627                                    <1>
1628                                    <1>     ; 19/11/2020
1629                                    <1>     ; case 3
1630                                    <1>     ; If '_set_mode' procedure is called for video mode 3
```

```
1631                            <1>       ;     while video mode is not 3 and if there is video
1632                            <1>       ;     page backup for video mode 3, all (of 8) mode 3
1633                            <1>       ;     video pages will be restored from 98000h.
1634                            <1>
1635 00001BAD A2[DA6F0000]     <1>       mov   [CRT_MODE], al  ; save mode in global variable
1636                            <1> _sm_3:
1637                            <1>       ; 30/07/2016
1638                            <1>       ; 26/07/2016
1639                            <1>       ; 25/07/2016
1640                            <1>       ; set_mode_vga:
1641                            <1>       ; 18/07/2016
1642                            <1>       ; 14/07/2016
1643                            <1>       ; 09/07/2016
1644                            <1>       ; 04/07/2016
1645                            <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
1646                            <1>       ; /// video mode 13h ///
1647                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
1648                            <1>       ; vgabios-0.7a (2011)
1649                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
1650                            <1>       ; 'vgabios.c', 'vgatables.h'
1651                            <1>       ;
1652                            <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
1653                            <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
1654                            <1>       ;
1655 00001BB2 88C4              <1>       mov   ah, al
1656 00001BB4 B910000000        <1>       mov   ecx, vga_mode_count
1657 00001BB9 BE[F66F0000]      <1>       mov   esi, vga_modes
1658 00001BBE 31DB              <1>       xor   ebx, ebx
1659                            <1> _sm_4:
1660 00001BC0 AC                <1>       lodsb
1661 00001BC1 38C4              <1>       cmp   ah, al
1662 00001BC3 7406              <1>       je    short _sm_5
1663 00001BC5 FEC3              <1>       inc   bl
1664 00001BC7 E2F7              <1>       loop  _sm_4
1665                            <1>
1666                            <1>       ; UNIMPLEMENTED VIDEO MODE !
1667                            <1>       ;xor  eax, eax
1668                            <1>         ;mov [video_eax], eax ; 0
1669                            <1>
1670                            <1>       ; 26/11/2020
1671 00001BC9 F9                <1>       stc   ; unimplemented video mode ! (cf=1)
1672                            <1>
1673 00001BCA C3                <1>       retn
1674                            <1>
1675                            <1> ;-----      eBX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
1676                            <1>
1677                            <1> _sm_5:      ; 25/07/2016
1678                            <1>       ;mov  esi, ebx
1679                            <1>       ;add  esi, vga_memmodel
1680                            <1>       ;mov  al, [esi]
1681                            <1>       ; 19/11/2020
1682 00001BCB 8A83[46700000]    <1>       mov   al, [ebx+vga_memmodel]
1683 00001BD1 A2[C6960100]      <1>       mov   [VGA_MTYPE], al
1684                            <1>
1685 00001BD6 89DF              <1>       mov   edi, ebx
1686 00001BD8 81C7[56700000]    <1>       add   edi, vga_dac_s
1687 00001BDE C0E302            <1>       shl   bl, 2 ; byte -> dword
1688 00001BE1 81C3[06700000]    <1>       add   ebx, vga_mode_tbl_ptr
1689                            <1>
1690                            <1>       ;mov  dword [VGA_BASE], 0B8000h
1691                            <1>       ;cmp  ah, 0Dh ; [CRT_MODE]
1692                            <1>       ;jb   short M9
1693                            <1>       ;mov  dword [VGA_BASE], 0A0000h
1694                            <1> ;M9:
1695 00001BE7 8B33              <1>       mov   esi, [ebx]
1696 00001BE9 89F3              <1>       mov   ebx, esi
1697 00001BEB 83C614            <1>       add   esi, vga_p_cm_pos ; ebx + 20
1698 00001BEE 668B06            <1>       mov   ax, [esi]       ; get the cursor mode from the table
1699 00001BF1 66A3[F36F0000]    <1>       mov   [CURSOR_MODE], ax ; save cursor mode (initial value)
1700                            <1>       ; al = 6, ah = 7
1701                            <1>       ; al = 0Dh, ah = 0Eh ; 25/07/2016
1702 00001BF7 E8A8020000        <1>       call  cursor_shape_fix
1703                            <1>       ; al = 14, ah = 15 (If [CHAR_HEIGHT] = 16)
1704 00001BFC 668906            <1>       mov   [esi], ax
1705                            <1>
1706 00001BFF 56                <1>       push  esi ; *
1707                            <1>
1708 00001C00 8A25[E16F0000]    <1>       mov   ah, [VGA_MODESET_CTL]
1709 00001C06 80E408            <1>       and   ah, 8 ; default palette loading ?
1710 00001C09 7524              <1>       jnz   short _sm_6
1711 00001C0B 66BAC603          <1>       mov   dx, 3C6h ; VGAREG_PEL_MASK (DAC mask register)
1712 00001C0F B0FF              <1>       mov   al, 0FFh ; PEL mask
1713 00001C11 EE                <1>       out   dx, al
1714 00001C12 8A27              <1>       mov   ah, [edi] ; DAC model (selection number)
1715 00001C14 E87E100000        <1>       call  load_dac_palette
1716                            <1>       ; ecx = 0
1717 00001C19 F605[E16F0000]02  <1>       test  byte [VGA_MODESET_CTL], 2 ; gray scale summing
1718 00001C20 740D              <1>       jz    short _sm_6
1719 00001C22 53                <1>       push  ebx
1720 00001C23 29DB              <1>       sub   ebx, ebx ; sub bl, bl
1721 00001C25 66B90001          <1>         mov     cx, 256
1722 00001C29 E8BC100000        <1>       call  gray_scale_summing
1723 00001C2E 5B                <1>       pop   ebx
1724                            <1> _sm_6:
1725                            <1>       ; Reset Attribute Ctl flip-flop
1726 00001C2F 66BADA03          <1>       mov   dx, 3DAh ; VGAREG_ACTL_RESET
1727 00001C33 EC                <1>       in    al, dx
1728                            <1>       ; Set Attribute Ctl
1729 00001C34 89DE              <1>       mov   esi, ebx ; addr of params tbl for selected mode
1730 00001C36 83C623            <1>       add   esi, 35  ; actl regs
1731 00001C39 30E4              <1>       xor   ah, ah ; 0
1732 00001C3B 66BAC003          <1>       mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
1733                            <1> _sm_7:
1734 00001C3F 88E0              <1>       mov   al, ah
1735 00001C41 EE                <1>       out   dx, al ; index
```

```
1736 00001C42 AC                <1>       lodsb
1737                            <1>       ; DX = 3C0h = VGAREG_ACTL_WRITE_DATA
1738 00001C43 EE                <1>       out    dx, al ; value
1739 00001C44 FEC4              <1>       inc    ah
1740 00001C46 80FC14            <1>       cmp    ah, 20 ; number of actl registers
1741 00001C49 72F4              <1>       jb     short _sm_7
1742                            <1>       ;
1743 00001C4B 88E0              <1>       mov    al, ah ; 20
1744 00001C4D EE                <1>       out    dx, al ; index
1745 00001C4E 28C0              <1>       sub    al, al ; 0
1746 00001C50 EE                <1>       out    dx, al ; value
1747                            <1>       ;
1748                            <1>       ; Set Sequencer Ctl
1749 00001C51 89DE              <1>       mov    esi, ebx ; addr of params tbl for selected mode
1750 00001C53 83C605            <1>       add    esi, 5 ; sequ regs
1751                            <1>       ;
1752 00001C56 66BAC403          <1>       mov    dx, 3C4h  ; VGAREG_SEQU_ADDRESS
1753 00001C5A EE                <1>       out    dx, al ; 0
1754 00001C5B 6642              <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
1755 00001C5D B003              <1>       mov    al, 3
1756 00001C5F EE                <1>       out    dx, al
1757 00001C60 B401              <1>       mov    ah, 1
1758                            <1> _sm_8:
1759 00001C62 88E0              <1>       mov    al, ah
1760                            <1>       ;mov   dx, 3C4h ; VGAREG_SEQU_ADDRESS
1761 00001C64 664A              <1>       dec    dx
1762 00001C66 EE                <1>       out    dx, al ; index
1763 00001C67 AC                <1>       lodsb
1764 00001C68 6642              <1>       inc    dx ; 3C5h ; VGAREG_SEQU_DATA
1765 00001C6A EE                <1>       out    dx, al
1766 00001C6B 80FC04            <1>       cmp    ah, 4 ; number of sequ regs
1767 00001C6E 7304              <1>       jnb    short _sm_9
1768 00001C70 FEC4              <1>       inc    ah
1769 00001C72 EBEE              <1>       jmp    short _sm_8
1770                            <1> _sm_9:
1771                            <1>       ; Set Grafx Ctl
1772 00001C74 89DE              <1>       mov    esi, ebx ; addr of params tbl for selected mode
1773 00001C76 83C637            <1>       add    esi, 55 ; grdc regs
1774 00001C79 30E4              <1>       xor    ah, ah ; 0
1775                            <1> _sm_10:
1776 00001C7B 88E0              <1>       mov    al, ah
1777 00001C7D 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
1778 00001C81 EE                <1>       out    dx, al
1779 00001C82 AC                <1>       lodsb
1780 00001C83 6642              <1>       inc    dx ; 3CFh ; VGAREG_GRDC_DATA
1781 00001C85 EE                <1>       out    dx, al
1782 00001C86 FEC4              <1>       inc    ah
1783 00001C88 80FC09            <1>       cmp    ah, 9 ; number of grdc regs
1784 00001C8B 72EE              <1>       jb     short _sm_10
1785                            <1>       ;
1786                            <1>       ; Disable CRTC write protection
1787 00001C8D 66BAD403          <1>       mov    dx, 3D4h  ; VGAREG_VGA_CRTC_ADDRESS
1788                            <1>       ;mov   al, 11h
1789                            <1>       ;our   dx, al
1790                            <1>       ;inc   dx
1791                            <1>       ;sub   al, al
1792                            <1>       ;out   dx, al
1793 00001C91 66B81100          <1>       mov    ax, 11h
1794 00001C95 66EF              <1>       out    dx, ax
1795 00001C97 89DE              <1>       mov    esi, ebx ; addr of params tbl for selected mode
1796 00001C99 83C60A            <1>       add    esi, 10 ; crtc regs
1797                            <1>       ; ah = 0
1798                            <1> _sm_11:
1799 00001C9C 88E0              <1>       mov    al, ah
1800                            <1>       ; dx = 3D4h = VGAREG_VGA_CRTC_ADDRESS
1801 00001C9E EE                <1>       out    dx, al ; index
1802 00001C9F AC                <1>       lodsb
1803 00001CA0 6642              <1>       inc    dx  ; VGAREG_VGA_CRTC_ADDRESS + 1
1804 00001CA2 EE                <1>       out    dx, al ; value
1805 00001CA3 80FC18            <1>       cmp    ah, 24 ; number of crtc registers - 1
1806 00001CA6 7306              <1>       jnb    short _sm_12
1807 00001CA8 FEC4              <1>       inc    ah
1808 00001CAA 664A              <1>       dec    dx ; 3D4h
1809 00001CAC EBEE              <1>       jmp    short _sm_11
1810                            <1> _sm_12:
1811                            <1>       ; Set the misc register
1812 00001CAE 66BACC03          <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
1813 00001CB2 8A4309            <1>       mov    al, [ebx+9] ; misc reg
1814 00001CB5 EE                <1>       out    dx, al
1815                            <1>       ;
1816                            <1>       ; Enable video
1817 00001CB6 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
1818 00001CBA B020              <1>       mov    al, 20h
1819 00001CBC EE                <1>         out     dx, al   ; set bit 5 to 1
1820 00001CBD 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
1821 00001CC1 EC                <1>       in     al, dx
1822                            <1>       ;
1823                            <1>       ; 17/11/2020
1824                            <1>       ;cmp   byte [noclearmem], 0
1825                            <1>         ;ja  short _sm_15
1826                            <1>
1827 00001CC2 F605[AF960100]80  <1>       test   byte [noclearmem], 80h
1828 00001CC9 7540              <1>       jnz    short _sm_15
1829                            <1>
1830                            <1>       ; 29/07/2016
1831 00001CCB 31C0              <1>       xor    eax, eax
1832 00001CCD B900400000        <1>       mov    ecx, 4000h ; 16K words (32K)
1833 00001CD2 803D[C6960100]02  <1>       cmp     byte [VGA_MTYPE], 2  ; CTEXT, MTEXT, CGA
1834 00001CD9 7715              <1>       ja     short _sm_14    ; no ? (0A0000h)
1835 00001CDB BF00800B00        <1>       mov    edi, 0B8000h
1836 00001CE0 7409              <1>       je     short _sm_13 ; CGA graphics mode
1837                            <1>       ; 08/08/2016
1838 00001CE2 A3[C2960100]      <1>       mov    [VGA_INT43H], eax ; 0 ; default font
1839 00001CE7 66B82007          <1>       mov    ax, 0720h    ; CGA text mode
1840                            <1> _sm_13:
```

```
1841 00001CEB F366AB             <1>       rep    stosw
1842 00001CEE EB1B               <1>       jmp    short _sm_15
1843                             <1>
1844                             <1> _sm_14:
1845 00001CF0 BF00000A00         <1>       mov    edi, 0A0000h
1846                             <1>       ; ecx = 16384 dwords (64K)
1847 00001CF5 66BAC403           <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
1848 00001CF9 B002               <1>       mov    al, 2
1849 00001CFB EE                 <1>       out    dx, al
1850                             <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
1851 00001CFC 6642               <1>       inc    dx
1852 00001CFE EC                 <1>       in     al, dx ; mmask
1853 00001CFF 6650               <1>       push   ax
1854 00001D01 B00F               <1>       mov    al, 0Fh ; all planes
1855 00001D03 EE                 <1>       out    dx, al
1856 00001D04 30C0               <1>       xor    al, al ; 0
1857 00001D06 F3AB               <1>       rep    stosd  ; ecx = 163684 (64K)
1858 00001D08 6658               <1>       pop    ax
1859 00001D0A EE                 <1>       out    dx, al  ; mmask
1860                             <1> _sm_15:
1861                             <1>       ; ebx = addr of params tbl for selected mode
1862                             <1>       ; 10/08/2016
1863 00001D0B 668B03             <1>       mov    ax, [ebx] ; num of columns, 'twidth'
1864 00001D0E A2[DC6F0000]       <1>       mov    [CRT_COLS], al
1865                             <1>       ;; 26/07/2016
1866                             <1>       ;; CRTC_ADDRESS = 3D4h (always)
1867                             <1>       ;mov   ah, [ebx+1] ; num of rows, 'theightm1'
1868 00001D13 FEC4               <1>       inc    ah ; 09/07/2016
1869 00001D15 8825[E26F0000]     <1>       mov    [VGA_ROWS], ah
1870                             <1>       ; 10/08/2016
1871 00001D1B 8A4302             <1>       mov    al, [ebx+2]
1872 00001D1E A2[DE6F0000]       <1>       mov    [CHAR_HEIGHT], al
1873                             <1>       ; 29/07/2016
1874                             <1>       ; length of regen buffer in bytes
1875 00001D23 668B4B03           <1>       mov    cx, [ebx+3] ; 'slength_l'
1876 00001D27 66890D[B0960100]   <1>       mov    [CRT_LEN], cx
1877                             <1>       ;
1878                             <1>       ; 27/07/2016
1879 00001D2E 30E4               <1>       xor    ah, ah
1880 00001D30 A0[468A0100]       <1>       mov    al, [ACTIVE_PAGE] ; may be > 0 for mode 3
1881                             <1>       ;mul   word [CRT_LEN] ; 4096 for mode 3
1882 00001D35 66F7E1             <1>       mul    cx ; 29/07/2016
1883 00001D38 66A3[348A0100]     <1>       mov    [CRT_START], ax
1884                             <1>       ;
1885 00001D3E B060               <1>       mov    al, 60h
1886                             <1>       ;cmp   byte [noclearmem], 0
1887                             <1>       ;jna   short _sm_16
1888                             <1>       ;add   al, 80h
1889 00001D40 0A05[AF960100]     <1>       or     al, [noclearmem] ; 17/11/2020
1890                             <1> _sm_16:
1891 00001D46 A2[DF6F0000]       <1>       mov    [VGA_VIDEO_CTL], al
1892 00001D4B C605[E06F0000]F9   <1>       mov    byte [VGA_SWITCHES], 0F9h
1893 00001D52 8025[E16F0000]7F   <1>       and    byte [VGA_MODESET_CTL], 7Fh
1894                             <1>
1895 00001D59 5E                 <1>       pop    esi ; *
1896                             <1>
1897                             <1>       ; 26/07/2016
1898                             <1>       ; 07/07/2016
1899 00001D5A 668B0D[F36F0000]   <1>       mov    cx, [CURSOR_MODE] ; restore cursor mode (initial value)
1900 00001D61 66870E             <1>       xchg   cx, [esi] ; cl = start line, ch = end line
1901                             <1>                        ; reset to initial value
1902 00001D64 86E9               <1>       xchg   ch, cl  ; ch = start line, cl = end line
1903 00001D66 66890D[F36F0000]   <1>       mov    [CURSOR_MODE], cx ; save (fixed) cursor mode
1904                             <1>
1905                             <1>       ; 27/07/2016
1906 00001D6D 803D[C6960100]02   <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
1907 00001D74 7317               <1>       jnb    short _sm_17
1908                             <1>
1909                             <1>       ; Set cursor shape
1910                             <1>       ;mov   cx, 0607h
1911                             <1>       ;call  _set_ctype
1912                             <1>
1913                             <1>       ; 29/07/2016
1914 00001D76 B40A               <1>       mov    ah, 10 ; 6845 register for cursor set
1915 00001D78 E84D060000         <1>       call   m16    ; output cx register
1916                             <1>
1917                             <1>       ; 25/07/2016
1918 00001D7D 803D[DA6F0000]03   <1>         cmp     byte [CRT_MODE], 03h
1919 00001D84 7507               <1>       jne    short _sm_17
1920                             <1>       ; 26/07/2016
1921                             <1>
1922 00001D86 A0[468A0100]       <1>       mov    al, [ACTIVE_PAGE]
1923 00001D8B EB0B               <1>       jmp    short _sm_18
1924                             <1> _sm_17:
1925                             <1>       ; Set cursor pos for page 0..7
1926                             <1>       ;sub   ax, ax ; eax = 0
1927 00001D8D 29C0               <1>       sub    eax, eax ; 17/11/2020
1928 00001D8F BF[368A0100]       <1>       mov    edi, CURSOR_POSN
1929 00001D94 AB                 <1>       stosd
1930 00001D95 AB                 <1>       stosd
1931 00001D96 AB                 <1>       stosd
1932 00001D97 AB                 <1>       stosd
1933                             <1>       ;; Set active page 0
1934                             <1>       ;mov   [ACTIVE_PAGE], al ; 0
1935                             <1> _sm_18:
1936                             <1>       ; 29/07/2016
1937 00001D98 803D[C6960100]02   <1>       cmp    byte [VGA_MTYPE], 2 ; CTEXT, MTEXT
1938 00001D9F 737F               <1>         jnb  _sm_23
1939                             <1>
1940                             <1>       ;cmp   byte [CHAR_HEIGHT], 16
1941                             <1>       ;je    short _sm_19
1942                             <1>
1943                             <1>       ;; copy and activate 8x16 font
1944                             <1>
1945                             <1>       ; 26/07/2016
```

```
1946 00001DA1 B004                <1>       mov    al, 04h
1947                              <1>       ;sub   bl, bl
1948                              <1>       ; AX = 1104H ; Load ROM 8x16 Character Set
1949                              <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
1950 00001DA3 E8E8160000          <1>       call   load_text_8_16_pat
1951                              <1>
1952                              <1>       ; video_func_1103h:
1953                              <1>       ; biosfn_set_text_block_specifier:
1954                              <1>       ; BL = font block selector code
1955                              <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
1956                              <1>       ; (It is as BL = 0 for TRDOS 386)
1957 00001DA8 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
1958                              <1>       ;;mov  ah, bl
1959                              <1>       ;sub   ah, ah ; 0
1960                              <1>       ;mov   al, 03h
1961                              <1>       ; 19/11/2020
1962 00001DAC 66B80300           <1>       mov    ax, 03h
1963 00001DB0 66EF                <1>       out    dx, ax
1964                              <1> _sm_19:
1965                              <1>       ; 29/07/2016
1966                              <1>       ; 26/07/2016
1967                              <1>       ; 24/06/2016
1968                              <1>       ;mov   edi, 0B8000h
1969                              <1>       ;mov   cx, 4000h ; 16K words (32K)
1970                              <1>       ;
1971 00001DB2 30C0                <1>       xor    al, al
1972 00001DB4 3805[AD960100]      <1>         cmp     [p_crt_mode], al ; 0
1973 00001DBA 7705                <1>         ja      short _sm_20 ; 03h, 80h or 83h
1974                              <1>
1975                              <1>       ; case 1 - 19/11/2020
1976                              <1>       ;
1977                              <1>       ; If [pc_crt_mode] = 0, that means, previous mode is 03h
1978                              <1>       ; and current mode is not 03h (case 1)
1979                              <1>
1980                              <1>       ; 30/07/2016
1981                              <1>       ; 24/06/2016
1982                              <1>       ; TRDOS 386 (TRDOS v2) 'set mode' modification
1983                              <1>       ; (for multiscreen feature):
1984                              <1>       ; If '_set_mode' procedure is called for video mode 3
1985                              <1>       ;     while video mode is 3, video page will be cleared
1986                              <1>       ;     and cursor position of video page will be reset.
1987                              <1>       ; If '_set_mode' procedure is called for a video mode
1988                              <1>       ;     except video mode 3, while current video mode
1989                              <1>       ;     is 3, all video pages of mode 3 will be copied
1990                              <1>       ;     to 98000h address as backup, before mode change.
1991                              <1>       ; If '_set_mode' procedure is called for video mode 3
1992                              <1>       ;     while video mode is not 3 and if there is video
1993                              <1>       ;     page backup for video mode 3, all (of 8) mode 3
1994                              <1>       ;     video pages will be restored from 98000h.
1995                              <1>
1996                              <1>       ; 19/11/2020
1997                              <1>       ;mov   [ACTIVE_PAGE], al ; 0
1998                              <1>
1999                              <1>       ; Here,
2000                              <1>       ; video memory already cleared if [noclearmem] <> 80h
2001                              <1>
2002                              <1>       ;mov   ax, 0720h
2003                              <1>       ;mov   cx, 4000h ; 16K words (32K)
2004                              <1>       ;mov   edi, 0B8000h
2005                              <1>       ;rep   stosw
2006                              <1>       ;sub   al, al
2007                              <1>
2008                              <1>       ;jmp   short _sm_23
2009                              <1>
2010                              <1>       ; Set hardware side for the new active video page
2011                              <1>
2012 00001DBC E9FB010000          <1>       jmp    _set_active_page ; 19/11/2020
2013                              <1>
2014                              <1> _sm_20:
2015                              <1>       ; 19/11/2020
2016                              <1>       ; case 2 or case 3 or case 4 - 19/11/2020
2017                              <1>
2018                              <1>       ; 19/11/2020
2019 00001DC1 803D[DA6F0000]03    <1>       cmp    byte [CRT_MODE], 3  ; new video mode
2020 00001DC8 754F                <1>       jne    short _sm_22 ; al = 0 (& video mode <> 03h)
2021                              <1>                           ; case 4 - 19/11/2020
2022                              <1>                           ; ([p_crt_mode] = 83h)
2023                              <1>
2024                              <1>       ; case 2 or case 3 - 19/11/2020
2025                              <1>
2026                              <1>       ;movzx ebx, byte [ACTIVE_PAGE]
2027                              <1>       ; 19/11/2020
2028 00001DCA A0[AE960100]        <1>       mov    al, [p_crt_page] ; previous mode 3 active page
2029                              <1>       ;movzx ebx, al
2030                              <1>       ;shl   bl, 1 ; * 2
2031                              <1>       ;add   ebx, CURSOR_POSN
2032                              <1>
2033                              <1>       ; 29/07/2016
2034 00001DCF F605[AD960100]7F    <1>       test    byte [p_crt_mode], 7Fh ; 83h or 80h or 03h
2035 00001DD6 740F                <1>       jz     short _sm_21 ; do not restore video pages
2036                              <1>                           ; case 2 - 19/11/2020
2037                              <1>       ; case 3 - 19/11/2020
2038                              <1>
2039                              <1>       ; ([p_crt_mode] = 03h)
2040                              <1>
2041                              <1>       ; New video mode is 3 while current video mode is not 3
2042                              <1>       ; (multi screen) video pages will be restored from 098000h
2043                              <1>
2044                              <1>       ; 19/11/2020
2045 00001DD8 A2[468A0100]        <1>       mov    [ACTIVE_PAGE], al ; current mode 3 active page
2046                              <1>
2047                              <1>       ; 12/12/2020
2048                              <1>       ;; restore video pages
2049                              <1>       ;mov   esi, 98000h ; 30/07/2016
2050                              <1>       ;mov   edi, 0B8000h
```

```
                                <1>       ;mov   cx, 2000h ; 8K dwords (32K)
                                <1>       ;rep   movsd
                                <1>       ;
                                <1>       ;; 19/11/2020
                                <1>       ;mov   [p_crt_mode], cl ; reset ('case 3' end condition)
                                <1>       ;
                                <1>       ;; restore cursor positions
                                <1>       ;mov   esi, cursor_pposn
                                <1>       ;mov   edi, CURSOR_POSN
                                <1>       ;;mov  ecx, 4 ; restore all cursor positions (16 bytes)
                                <1>       ;mov   cl, 4
                                <1>       ;rep   movsd
                                <1>
                                <1>       ; 12/12/2020
00001DDD E89C000000             <1>       call   _restore_mode3_multiscreen
                                <1>
                                <1>       ;jmp   short _sm_23 ; do not clear current video pages
                                <1>
                                <1>       ; 19/11/2020
00001DE2 E9D5010000             <1>       jmp    _set_active_page
                                <1>
                                <1> _sm_21:
                                <1>       ; 19/11/2020
                                <1>       ; case 2
                                <1>       ;
                                <1>       ; ([p_crt_mode] = 80h)
                                <1>       ;
                                <1>       ; User has requested to set video mode 3 again while
                                <1>       ; current video mode is 3.. that means, set mode 03h
                                <1>       ; parameters again and clear video page.
                                <1>       ; ('noclearmem' option effects the result)
                                <1>
                                <1>       ; 19/11/2020
00001DE7 F605[AF960100]80       <1>       test   byte [noclearmem], 80h
00001DEE 7529                   <1>       jnz    short _sm_22 ; 'do not clear video memory'
                                <1>                           ; continue with current text
                                <1>                           ; (user's option/choice)
                                <1>       ; clear video page
                                <1>       ;mov   cx, [CRT_LEN] ; 4096
                                <1>       ;shr   cx, 1 ; 2048 ; 16/11/2020
00001DF0 66B82007               <1>       mov    ax, 0720h
                                <1>       ; 26/11/2020
00001DF4 B900080000             <1>       mov    ecx, 2048 ; 4096/2
00001DF9 BF00800B00             <1>       mov    edi, 0B8000h ; [crt_base]
00001DFE 66033D[348A0100]       <1>       add    di, [CRT_START]
00001E05 F366AB                 <1>       rep    stosw  ; FILL THE REGEN BUFFER WITH BLANKS
                                <1>       ;
                                <1>       ; 19/11/2020
00001E08 A0[468A0100]           <1>       mov    al, [ACTIVE_PAGE] ; 0 to 7 (for video mode 3)
00001E0D 0FB6D8                 <1>       movzx  ebx, al
00001E10 D0E3                   <1>       shl    bl, 1
00001E12 66898B[368A0100]       <1>       mov    [ebx+CURSOR_POSN], cx ; reset cursor position
                                <1> _sm_22:
                                <1>       ;mov   [p_crt_mode], al ; 0 ; reset
                                <1>       ; 19/11/2020
                                <1>       ;and   byte [p_crt_mode], 3 ; 83h -> 3, 80h -> 0
00001E19 8025[AD960100]7F       <1>       and    byte [p_crt_mode], 7Fh ; 83h -> 3, 80h -> 0
                                <1> _sm_23:
                                <1>       ; al = video page number
                                <1>       ; [CRT_LEN] = length of regen buffer in bytes
                                <1>       ;call  _set_active_page
                                <1>       ; 16/11/2020
00001E20 E997010000             <1>       jmp    _set_active_page
                                <1>
                                <1> ;-----       NORMAL RETURN FROM ALL VIDEO RETURNS
                                <1>       ;retn
                                <1>
                                <1> save_mode3_multiscreen:
                                <1>       ; 12/12/2020 (TRDOS v2.0.3)
                                <1>       ; save mode 03h video pages and cursor positions
                                <1>       ;
                                <1>       ; Modified registers: ecx (=0), esi, edi
                                <1>
                                <1>       ; 12/12/2020
                                <1>       ; moved here from '_set_mode'
                                <1>       ; 03/07/2016
                                <1>       ; save video pages
00001E25 BE00800B00             <1>       mov    esi, 0B8000h
00001E2A BF00800900             <1>       mov    edi, 98000h ; 30/07/2016
00001E2F B900200000             <1>       mov    ecx, (0B8000h-0B0000h)/4
00001E34 F3A5                   <1>       rep    movsd
                                <1>
00001E36 C605[AD960100]03       <1>       mov    byte [p_crt_mode], 3 ; previous mode, backup sign
                                <1>       ; 26/11/2020
00001E3D 860D[468A0100]         <1>       xchg   cl, [ACTIVE_PAGE]
00001E43 880D[AE960100]         <1>       mov    [p_crt_page], cl  ; save as previous active page
                                <1>
                                <1>       ; save cursor positions
00001E49 BE[368A0100]           <1>       mov    esi, CURSOR_POSN
00001E4E BF[B2960100]           <1>       mov    edi, cursor_pposn ; cursor positions backup
00001E53 B104                   <1>       mov    cl, 4
00001E55 F3A5                   <1>       rep    movsd
00001E57 C3                     <1>       retn
                                <1>
                                <1> restore_mode3_multiscreen:
                                <1>       ; 12/12/2020 (TRDOS v2.0.3)
                                <1>       ; restore mode 03h video pages and cursor positions
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;    settings from the last 'save_mode3_multiscreen'
                                <1>       ;
                                <1>       ; Output:
                                <1>       ;    AL = active video page = [ACTIVE_PAGE]
                                <1>       ;
                                <1>       ; Modified registers: al, ecx (=0), esi, edi
```

```
2156                                    <1>
2157 00001E58 A0[AE960100]              <1>       mov   al, [p_crt_page] ; previous mode 3 active page
2158 00001E5D A2[468A0100]              <1>       mov   [ACTIVE_PAGE], al ; current mode 3 active page
2159                                    <1>
2160                                    <1>       ; 12/12/2020
2161                                    <1>       ; moved here from 'vesa_vbe3_pmi'
2162                                    <1>
2163                                    <1>       ; 07/12/2020
2164                                    <1>       ; restore CRT_START according to ACTIVE_PAGE
2165                                    <1>       ;mov  [CRT_START], cx ; 0
2166                                    <1>       ; 12/12/2020
2167 00001E62 66C705[348A0100]00-       <1>       mov   word [CRT_START], 0
2167 00001E6A 00                        <1>
2168                                    <1>
2169                                    <1>       ; check active page and set it again if it is not 0
2170 00001E6B 08C0                      <1>       or    al, al
2171                                    <1>       ;;jz  short vbe3_pmi_7
2172                                    <1>       ;jz   short _restore_mode3_multiscreen
2173 00001E6D 740F                      <1>       jz    short r_m3_ms_1
2174 00001E6F 88C1                      <1>       mov   cl, al
2175                                    <1> ;vbe3_pmi_5:
2176                                    <1> r_m3_ms_0:
2177 00001E71 668105[348A0100]00-       <1>       add   word [CRT_START], 4096
2177 00001E79 10                        <1>
2178 00001E7A FEC9                      <1>       dec   cl
2179                                    <1>       ;jnz  short vbe3_pmi_5
2180 00001E7C 75F3                      <1>       jnz   short r_m3_ms_0
2181                                    <1> r_m3_ms_1:
2182                                    <1>       ; 12/12/2020
2183                                    <1>       ; moved here from '_set_mode'
2184                                    <1> _restore_mode3_multiscreen:
2185                                    <1>       ; Modified registers: ecx, esi, edi
2186                                    <1>
2187                                    <1>       ; restore video pages
2188 00001E7E BE00800900                <1>       mov   esi, 98000h ; 30/07/2016
2189 00001E83 BF00800B00                <1>       mov   edi, 0B8000h
2190                                    <1>       ;mov  cx, 2000h ; 8K dwords (32K)
2191 00001E88 B900200000                <1>       mov   ecx, 2000h
2192 00001E8D F3A5                      <1>       rep   movsd
2193                                    <1>
2194                                    <1>       ; 19/11/2020
2195 00001E8F 880D[AD960100]            <1>       mov   [p_crt_mode], cl ; reset ('case 3' end condition)
2196                                    <1>
2197                                    <1>       ; restore cursor positions
2198 00001E95 BE[B2960100]              <1>       mov   esi, cursor_pposn
2199 00001E9A BF[368A0100]              <1>       mov   edi, CURSOR_POSN
2200                                    <1>       ;mov  ecx, 4 ; restore all cursor positions (16 bytes)
2201 00001E9F B104                      <1>       mov   cl, 4
2202 00001EA1 F3A5                      <1>       rep   movsd
2203 00001EA3 C3                        <1>       retn
2204                                    <1>
2205                                    <1> cursor_shape_fix:
2206                                    <1>       ; 07/07/2016
2207                                    <1>       ; (Cursor start and cursor end line values -6,7-
2208                                    <1>       ; will be fixed depending on character height)
2209                                    <1>       ;
2210                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
2211                                    <1>       ; vgabios-0.7a (2011)
2212                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
2213                                    <1>       ; 'vgabios.c', ' biosfn_set_cursor_shape (CH,CL)'
2214                                    <1>       ;
2215                                    <1>       ; INPUT ->
2216                                    <1>       ;     AL = cursor start line (=6)
2217                                    <1>       ;     AH = cursor end line (=7)
2218                                    <1>       ; OUTPUT ->
2219                                    <1>       ;     AL = cursor start line (=14)
2220                                    <1>       ;     AH = cursor end line (=15)
2221                                    <1>       ;
2222                                    <1>       ;; if((modeset_ctl&0x01)&&(cheight>8)&&(CL<8)&&(CH<0x20))
2223                                    <1>
2224                                    <1>       ;test  byte [VGA_MODESET_CTL], 1 ; VGA active
2225                                    <1>       ;jz   short csf_3
2226 00001EA4 803D[DE6F0000]08          <1>       cmp   byte [CHAR_HEIGHT], 8
2227 00001EAB 7649                      <1>       jna   short csf_3
2228 00001EAD 80FC08                    <1>       cmp   ah, 8
2229 00001EB0 7344                      <1>       jnb   short csf_3
2230 00001EB2 3C20                      <1>       cmp   al, 20h
2231 00001EB4 7340                      <1>       jnb   short csf_3
2232                                    <1>       ;
2233 00001EB6 6650                      <1>       push  ax
2234                                    <1>       ; {
2235                                    <1>       ; if(CL!=(CH+1))
2236 00001EB8 FEC0                      <1>       inc   al
2237 00001EBA 38C4                      <1>       cmp   ah, al   ; ah != al + 1
2238 00001EBC 740F                      <1>         je      short csf_1
2239                                    <1>       ; CH = ((CH+1) * cheight / 8) -1;
2240 00001EBE 8A25[DE6F0000]            <1>       mov   ah, [CHAR_HEIGHT]
2241 00001EC4 F6E4                      <1>       mul   ah
2242 00001EC6 C0E803                    <1>       shr   al, 3 ; / 8
2243 00001EC9 FEC8                      <1>       dec   al ; - 1
2244 00001ECB EB0E                      <1>       jmp   short csf_2
2245                                    <1> csf_1:
2246                                    <1>       ; }
2247                                    <1>       ; else       ; ah = al + 1
2248                                    <1>       ; {
2249 00001ECD FEC4                      <1>       inc   ah     ; ah = ah + 1
2250                                    <1>       ; CH = ((CL+1) * cheight / 8) - 2;
2251 00001ECF A0[DE6F0000]              <1>       mov   al, [CHAR_HEIGHT]
2252 00001ED4 F6E4                      <1>       mul   ah
2253 00001ED6 C0E803                    <1>       shr   al, 3 ; / 8
2254 00001ED9 2C02                      <1>       sub   al, 2 ; - 2
2255                                    <1>       ; al = 14 (if [CHAR_HEIGHT] = 16)
2256                                    <1> csf_2:
2257 00001EDB 880424                    <1>       mov   [esp], al
2258 00001EDE 8A642401                  <1>       mov   ah, [esp+1]
```

```
2259                                    <1>       ; CL = ((CL+1) * cheight / 8) - 1;
2260 00001EE2 FEC4                     <1>       inc    ah
2261 00001EE4 A0[DE6F0000]             <1>       mov    al, [CHAR_HEIGHT]
2262 00001EE9 F6E4                     <1>       mul    ah
2263 00001EEB C0E803                   <1>       shr    al, 3 ; / 8
2264 00001EEE FEC8                     <1>       dec    al ; - 1
2265 00001EF0 88442401                 <1>       mov    [esp+1], al
2266                                    <1>       ; ah = 15 (if [CHAR_HEIGHT] = 16)
2267                                    <1>       ;
2268 00001EF4 6658                     <1>       pop    ax
2269                                    <1> csf_3:
2270 00001EF6 C3                       <1>       retn
2271                                    <1>
2272                                    <1> SET_CTYPE:
2273                                    <1>       ; 12/09/2016
2274                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2275 00001EF7 803D[DA6F0000]07         <1>       cmp    byte [CRT_MODE], 7
2276 00001EFE 0F875BFCFFFF             <1>       ja     VIDEO_RETURN ; 12/09/2016
2277 00001F04 E805000000               <1>       call   _set_ctype
2278 00001F09 E951FCFFFF               <1>         jmp     VIDEO_RETURN
2279                                    <1>
2280                                    <1> _set_ctype:
2281                                    <1>       ; 02/09/2014 (Retro UNIX 386 v1)
2282                                    <1>       ;
2283                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2284                                    <1>
2285                                    <1>       ; (CH) = BITS 4-0 = START LINE FOR CURSOR
2286                                    <1>       ;  ** HARDWARE WILL ALWAYS CAUSE BLINK
2287                                    <1>       ;  ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC BLINKING
2288                                    <1>       ;     OR NO CURSOR AT ALL
2289                                    <1>       ; (CL) = BITS 4-0 = END LINE FOR CURSOR
2290                                    <1>
2291                                    <1> ;--------------------------------------------
2292                                    <1> ; SET_CTYPE
2293                                    <1> ;     THIS ROUTINE SETS THE CURSOR VALUE
2294                                    <1> ; INPUT
2295                                    <1> ;     (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
2296                                    <1> ; OUTPUT
2297                                    <1> ;     NONE
2298                                    <1> ;--------------------------------------------
2299                                    <1>
2300                                    <1>       ; 07/07/2016
2301                                    <1>       ; Fixing cursor start and stop line depending on
2302                                    <1>       ; current character height (=16)
2303                                    <1>       ; (Note: Default/initial values are 6 and 7.
2304                                    <1>       ; If set values are 6 (start) & 7 (stop) and
2305                                    <1>       ; [CHAR_HEIGHT] = 16 :
2306                                    <1>       ; After fixing, start line will be 14, stop line
2307                                    <1>       ; will be 15.)
2308 00001F0E 6689C8                   <1>       mov    ax, cx
2309 00001F11 86C4                     <1>       xchg   al, ah
2310                                    <1>       ; AL = start line, AH = stop line
2311 00001F13 E88CFFFFFF               <1>       call   cursor_shape_fix
2312                                    <1>       ; AL = start line (fixed), AH = stop line (fixed)
2313 00001F18 6689C1                   <1>       mov    cx, ax
2314 00001F1B 86E9                     <1>       xchg   ch, cl
2315                                    <1>       ; CH = start line (fixed), CL = stop line (fixed)
2316                                    <1>       ;
2317 00001F1D B40A                     <1>       mov    ah, 10 ; 6845 register for cursor set
2318 00001F1F 66890D[F36F0000]         <1>       mov    [CURSOR_MODE], cx ; save in data area
2319                                    <1>       ;call  m16    ; output cx register
2320                                    <1>       ;retn
2321 00001F26 E99F040000               <1>         jmp     m16
2322                                    <1>
2323                                    <1> SET_CPOS:
2324                                    <1>       ; 12/09/2016
2325                                    <1>       ; 07/07/2016
2326                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2327 00001F2B 80FF07                   <1>       cmp    bh, 7 ; video page > 7 ; 07/07/2016
2328 00001F2E 0F872BFCFFFF             <1>       ja     VIDEO_RETURN
2329                                    <1>       ;
2330 00001F34 803D[DA6F0000]07         <1>       cmp    byte [CRT_MODE], 7
2331 00001F3B 770A                     <1>       ja     short vga_set_cpos ; 12/09/2016
2332 00001F3D E85D040000               <1>       call   _set_cpos
2333 00001F42 E918FCFFFF               <1>         jmp     VIDEO_RETURN
2334                                    <1>
2335                                    <1> vga_set_cpos:
2336                                    <1>       ; 12/09/2016
2337                                    <1>       ; 09/07/2016
2338                                    <1>       ; set cursor position
2339                                    <1>       ; NOTE: Hardware cursor position will not be set
2340                                    <1>       ;   in any VGA modes (>7)
2341                                    <1>       ;   But, cursor position will be saved into
2342                                    <1>       ;   [CURSOR_POSN].
2343                                    <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
2344                                    <1>       ;   (page 0) for all graphics modes.
2345                                    <1>
2346 00001F47 668915[368A0100]         <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
2347                                    <1>       ; 04/08/2016
2348                                    <1>       ;mov   bh, [ACTIVE_PAGE] ; = 0
2349                                    <1>       ;call  _set_cpos
2350 00001F4E E90CFCFFFF               <1>       jmp     VIDEO_RETURN
2351                                    <1>
2352                                    <1> READ_CURSOR:
2353                                    <1>       ; 12/09/2016
2354                                    <1>       ; 07/07/2016
2355                                    <1>       ; 12/05/2016
2356                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2357                                    <1>       ;
2358                                    <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2359                                    <1>
2360                                    <1> ;--------------------------------------------------
2361                                    <1> ; READ_CURSOR
2362                                    <1> ;     THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE
2363                                    <1> ;     845, FORMATS IT, AND SENDS IT BACK TO THE CALLER
```

```
2364                            <1> ; INPUT
2365                            <1> ;     BH - PAGE OF CURSOR
2366                            <1> ; OUTPUT
2367                            <1> ;     DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION
2368                            <1> ;     CX - CURRENT CURSOR MODE
2369                            <1> ;--------------------------------------------------------
2370                            <1>
2371                            <1>       ; BH = Video page number (0 to 7)
2372                            <1>
2373                            <1>       ; 07/07/2016
2374 00001F53 80FF07           <1>       cmp    bh, 7 ; video page > 7 (invalid)
2375 00001F56 7606             <1>       jna    short read_cursor_1
2376                            <1>       ; invalid video page (input)
2377 00001F58 31C9             <1>       xor    ecx, ecx ; 0
2378 00001F5A 31D2             <1>       xor    edx, edx ; 0
2379 00001F5C EB15             <1>       jmp    short read_cursor_2
2380                            <1> read_cursor_1:
2381                            <1>       ; 12/09/2016
2382 00001F5E 803D[DA6F0000]07 <1>       cmp    byte [CRT_MODE], 7 ; vga mode
2383 00001F65 7727             <1>       ja     short vga_get_cpos
2384                            <1>       ;
2385 00001F67 E815000000       <1>       call   get_cpos
2386 00001F6C 0FB70D[F36F0000] <1>       movzx  ecx, word [CURSOR_MODE]
2387                            <1> read_cursor_2:
2388 00001F73 5D               <1>       pop    ebp
2389 00001F74 5F               <1>       pop    edi
2390 00001F75 5E               <1>       pop    esi
2391 00001F76 5B               <1>       pop    ebx
2392 00001F77 58               <1>       pop    eax  ; DISCARD SAVED CX AND DX
2393 00001F78 58               <1>       pop    eax
2394 00001F79 A1[A0960100]     <1>       mov    eax, [video_eax] ; 12/05/2016
2395                            <1>       ;;15/01/2017
2396                            <1>       ;;mov  byte [intflg], 0 ; 07/01/2017
2397 00001F7E 1F               <1>       pop    ds
2398 00001F7F 07               <1>       pop    es
2399 00001F80 CF               <1>       iretd
2400                            <1>
2401                            <1> get_cpos:
2402                            <1>       ; 12/05/2016
2403                            <1>       ; 16/01/2016
2404                            <1>       ; BH = Video page number (0 to 7)
2405                            <1>       ;
2406 00001F81 D0E7             <1>       shl    bh, 1 ; WORD OFFSET
2407 00001F83 0FB6F7           <1>       movzx  esi, bh
2408 00001F86 0FB796[368A0100] <1>       movzx  edx, word [esi+CURSOR_POSN]
2409 00001F8D C3               <1>       retn
2410                            <1>
2411                            <1> vga_get_cpos:
2412                            <1>       ; 12/09/2016
2413                            <1>       ; get cursor position (vga)
2414 00001F8E 0FB715[368A0100] <1>       movzx  edx, word [CURSOR_POSN] ; cursor pos for pg 0
2415 00001F95 31C9             <1>       xor    ecx, ecx ; Cursor Mode = 0 (invalid)
2416 00001F97 EBDA             <1>       jmp     short read_cursor_2
2417                            <1>
2418                            <1> ACT_DISP_PAGE:
2419                            <1>       ; 07/07/2016
2420                            <1>       ; 26/06/2016
2421                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2422                            <1>       ;
2423                            <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2424                            <1>       ;
2425                            <1> ;--------------------------------------------------------
2426                            <1> ; ACT_DISP_PAGE
2427                            <1> ;     THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING
2428                            <1> ;     THE FULL USE OF THE MEMORY SET ASIDE FOR THE VIDEO ATTACHMENT
2429                            <1> ; INPUT
2430                            <1> ;     AL HAS THE NEW ACTIVE DISPLAY PAGE
2431                            <1> ; OUTPUT
2432                            <1> ;     THE 6845 IS RESET TO DISPLAY THAT PAGE
2433                            <1> ;--------------------------------------------------------
2434                            <1>       ; 07/07/2016
2435 00001F99 3C07             <1>       cmp    al, 7  ; > 7 = invalid video page number
2436                            <1>       ;ja    VIDEO_RETURN
2437 00001F9B 7715             <1>         ja   short adp_2 ; 18/11/2020
2438                            <1>       ;cmp   byte [CRT_MODE], 3
2439                            <1>       ;je    short adp_1
2440                            <1>       ; 18/11/2020
2441 00001F9D 8A25[DA6F0000]   <1>       mov    ah, [CRT_MODE]
2442 00001FA3 80FC03           <1>       cmp    ah, 3
2443 00001FA6 7605             <1>       jna    short adp_1 ; mode 01h, 00h (01h), 02h (03h), 03h
2444 00001FA8 80FC07           <1>       cmp    ah, 7      ; mode 07h (03h)
2445 00001FAB 7505             <1>       jne    short adp_2
2446                            <1>       ;and   al, al
2447                            <1>         ;jnz VIDEO_RETURN
2448                            <1>       ;;sub  al, al ; 0 ; force to page 0
2449                            <1> adp_1:
2450 00001FAD E805000000       <1>       call   set_active_page
2451                            <1> adp_2:
2452 00001FB2 E9A8FBFFFF       <1>         jmp     VIDEO_RETURN
2453                            <1>
2454                            <1> set_active_page:   ; tty_sw
2455                            <1>       ; 09/12/2017
2456                            <1>       ; 26/07/2016
2457                            <1>       ; 26/06/2016
2458                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2459                            <1>       ; 30/06/2015
2460                            <1>       ; 04/03/2014  (act_disp_page --> tty_sw)
2461                            <1>       ; 10/12/2013
2462                            <1>       ; 04/12/2013
2463                            <1>       ;
2464 00001FB7 A2[468A0100]     <1>       mov    [ACTIVE_PAGE], al ; save active page value ; [ptty]
2465                            <1> _set_active_page:
2466                            <1>       ; 27/06/2015
2467 00001FBC 0FB6D8           <1>       movzx  ebx, al
2468                            <1>       ;
```

```
                                <1>       ;cbw   ; 07/09/2014 (ah=0)
00001FBF 28E4                   <1>       sub    ah, ah ; 09/12/2017
00001FC1 66F725[B0960100]       <1>       mul    word [CRT_LEN]  ; get saved length of regen buffer
                                <1>                              ; display page times regen length
                                <1>       ; 10/12/2013
00001FC8 66A3[348A0100]         <1>       mov    [CRT_START], ax ; save start address for later
00001FCE 6689C1                 <1>       mov    cx, ax ; start address to cx
                                <1> _M16:
                                <1>       ;sar   cx, 1
00001FD1 66D1E9                 <1>       shr    cx, 1  ; divide by 2 for 6845 handling
00001FD4 B40C                   <1>       mov    ah, 12 ; 6845 register for start address
00001FD6 E8EF030000             <1>       call   m16
                                <1>       ;sal   bx, 1
                                <1>       ; 01/09/2014
00001FDB D0E3                   <1>       shl    bl, 1  ; *2 for word offset
00001FDD 81C3[368A0100]         <1>       add    ebx, CURSOR_POSN
00001FE3 668B13                 <1>       mov    dx, [ebx] ; get cursor for this page
                                <1>       ; 16/01/2016
                                <1>       ;call  m18
                                <1>       ;retn
00001FE6 E9CB030000             <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
00001FEB 0FB605[DC6F0000]       <1>       movzx  eax, byte [CRT_COLS] ; 24/06/2016
00001FF2 F6E6                   <1>       mul    dh      ; row value
00001FF4 30F6                   <1>       xor    dh, dh   ; 0
00001FF6 6601D0                 <1>       add    ax, dx  ; add column value to the result
00001FF9 66D1E0                 <1>       shl    ax, 1  ; * 2 for attribute bytes
                                <1>              ; EAX = AX = OFFSET OF CHAR POSITION IN REGEN BUFFER
00001FFC 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>
00001FFD 0FB6CF                 <1>       movzx  ecx, bh ; video page number
00002000 89CE                   <1>       mov    esi, ecx
00002002 66D1E6                 <1>       shl    si, 1
00002005 668B96[368A0100]       <1>       mov    dx, [esi+CURSOR_POSN]
0000200C 740C                   <1>       jz     short p21
0000200E 6631F6                 <1>       xor    si, si
                                <1> p20:
00002011 660335[B0960100]       <1>       add    si, [CRT_LEN] ; 24/06/2016
                                <1>       ;add   si, 80*25*2 ; add length of buffer for one page
00002018 E2F7                   <1>       loop   p20
                                <1> p21:
0000201A 6621D2                 <1>       and    dx, dx
0000201D 7407                   <1>       jz     short p22
0000201F E8C7FFFFFF             <1>       call   position ; determine location in regen in page
00002024 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
00002026 66BADA03               <1>       mov    dx, 03DAh ; status port
                                <1>       ; cx = 0
0000202A 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
0000202B 38F5                   <1>         cmp     ch, dh
0000202D 0F872CFBFFFF           <1>         ja      VIDEO_RETURN
00002033 38D1                   <1>         cmp     cl, dl
00002035 0F8724FBFFFF           <1>         ja      VIDEO_RETURN
                                <1>         ;
0000203B E805000000             <1>         call    _scroll_up
00002040 E91AFBFFFF             <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>
00002045 E89C000000             <1>         call    test_line_count ; 16/01/2016
                                <1>
0000204A 8A25[DA6F0000]         <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
00002050 80FC04                 <1>         cmp     ah, 4
00002053 720A                   <1>         jb      short n0
00002055 80FC07                 <1>         cmp     ah, 7 ; TEST FOR BW CARD
                                <1>                       ;       (80x25 text, mono)
00002058 7405                   <1>         je      short n0 ; same with mode 3 for TRDOS 386
0000205A E942050000             <1>         jmp     GRAPHICS_UP
                                <1> n0:
                                <1>         ; 07/07/2016
0000205F 80FF07                 <1>         cmp     bh, 7 ; video page number
00002062 7606                   <1>         jna     short n1
00002064 8A3D[468A0100]         <1>         mov     bh, [ACTIVE_PAGE]
                                <1> n1:
0000206A 88DC                   <1>         mov     ah, bl ; attribute
0000206C 6650                   <1>         push    ax ; *
                                <1>         ;mov   esi, [CRT_BASE]
0000206E BE00800B00             <1>           mov     esi, 0B8000h
00002073 3A3D[468A0100]         <1>           cmp     bh, [ACTIVE_PAGE]
00002079 750B                   <1>         jne     short n2
                                <1>         ;
0000207B 66A1[348A0100]         <1>           mov     ax, [CRT_START]
00002081 6601C6                 <1>           add     si, ax
00002084 EB11                   <1>           jmp     short n4
                                <1> n2:
00002086 20FF                   <1>           and     bh, bh
00002088 740D                   <1>         jz      short n4
0000208A 88F8                   <1>         mov     al, bh
                                <1> n3:
0000208C 660335[B0960100]       <1>           add  si, [CRT_LEN]
00002093 FEC8                   <1>           dec  al
00002095 75F5                   <1>         jnz     short n3
                                <1> n4:
00002097 E85D000000             <1>         call    scroll_position ; 16/01/2016
0000209C 7420                   <1>           jz      short n6
                                <1>
0000209E 01CE                   <1>           add     esi, ecx ; from address for scroll
000020A0 88F5                   <1>         mov     ch, dh  ; #rows in block
000020A2 28C5                   <1>         sub     ch, al ; #rows to be moved
                                <1> n5:
000020A4 E894000000             <1>         call    n10 ; 16/01/2016
                                <1>
000020A9 51                     <1>           push ecx
000020AA 0FB60D[DC6F0000]       <1>         movzx  ecx, byte [CRT_COLS]
000020B1 00C9                   <1>         add     cl, cl
000020B3 01CE                   <1>           add  esi, ecx  ; next line
000020B5 01CF                   <1>           add  edi, ecx
000020B7 59                     <1>         pop     ecx
                                <1>
000020B8 FECD                   <1>         dec     ch      ; count of lines to move
000020BA 75E8                   <1>         jnz     short n5 ; row loop
                                <1>         ; ch = 0
000020BC 88C6                   <1>         mov     dh, al  ; #rows
                                <1> n6:
                                <1>         ; attribute in ah
000020BE B020                   <1>         mov     al, ' '       ; fill with blanks
                                <1> n7:
000020C0 E885000000             <1>         call    n11 ; 16/01/2016
                                <1>
000020C5 8A0D[DC6F0000]         <1>         mov     cl, [CRT_COLS]
000020CB 00C9                   <1>         add     cl, cl
000020CD 01CF                   <1>           add  edi, ecx
                                <1>
000020CF FECE                   <1>         dec     dh
000020D1 75ED                   <1>         jnz     short n7
                                <1> n16:
000020D3 3A3D[468A0100]         <1>         cmp     bh, [ACTIVE_PAGE]
000020D9 750A                   <1>         jne     short n8
                                <1>
                                <1>         ;cmp   byte [CRT_MODE], 7 ; is this the black and white card
                                <1>         ;je    short n8        ; if so, skip the mode reset
                                <1>
```

```
2679 000020DB A0[DB6F0000]        <1>       mov    al, [CRT_MODE_SET] ; get the value of mode set
2680 000020E0 66BAD803            <1>       mov    dx, 03D8h ; always set color card port
2681 000020E4 EE                  <1>       out    dx, al
2682                              <1> n8:
2683 000020E5 C3                  <1>       retn
2684                              <1>
2685                              <1> test_line_count:
2686                              <1>       ; 12/05/2016
2687                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2688                              <1>       ; 07/09/2014 (scroll_up)
2689 000020E6 08C0               <1>       or     al, al
2690 000020E8 740E               <1>       jz     short al_set2
2691 000020EA 6652               <1>       push   dx
2692 000020EC 28EE               <1>       sub    dh, ch  ; subtract upper row from lower row number
2693 000020EE FEC6               <1>       inc    dh     ; adjust difference by 1
2694 000020F0 38C6               <1>       cmp    dh, al       ; line count = amount of rows in window?
2695 000020F2 7502               <1>       jne    short al_set1 ; if not the we're all set
2696 000020F4 30C0               <1>       xor    al, al ; otherwise set al to zero
2697                              <1> al_set1:
2698 000020F6 665A               <1>       pop    dx
2699                              <1> al_set2:
2700 000020F8 C3                 <1>       retn
2701                              <1>
2702                              <1> scroll_position:
2703                              <1>       ; 26/06/2016
2704                              <1>       ; 30/01/2016
2705                              <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2706                              <1>       ; 07/09/2014 (scroll_up)
2707                              <1>
2708 000020F9 6652               <1>       push   dx
2709 000020FB 6689CA             <1>       mov    dx, cx ; now, upper left position in DX
2710 000020FE E8E8FEFFFF         <1>       call   position
2711 00002103 01C6               <1>       add    esi, eax
2712 00002105 89F7               <1>       mov    edi, esi
2713 00002107 665A               <1>       pop    dx     ; lower right position in DX
2714 00002109 6629CA             <1>       sub    dx, cx
2715 0000210C FEC6               <1>       inc    dh     ; dh = #rows
2716 0000210E FEC2               <1>       inc    dl     ; dl = #cols in block
2717 00002110 59                 <1>       pop    ecx    ; return address
2718 00002111 6658               <1>       pop    ax     ; * ; al = line count, ah = attribute
2719 00002113 51                 <1>       push   ecx    ; return address
2720 00002114 0FB7C8             <1>       movzx  ecx, ax
2721 00002117 8A25[DC6F0000]     <1>       mov    ah, [CRT_COLS]
2722 0000211D F6E4               <1>       mul    ah     ; determine offset to from address
2723 0000211F 6601C0             <1>       add    ax, ax  ; *2 for attribute byte
2724                              <1>       ;
2725 00002122 6650               <1>       push   ax     ; offset
2726 00002124 6652               <1>       push   dx
2727                              <1>       ;
2728                              <1>       ; 04/04/2014
2729 00002126 66BADA03           <1>       mov    dx, 3DAh ; guaranteed to be color card here
2730                              <1> n9:                   ; wait_display_enable
2731 0000212A EC                 <1>         in      al, dx   ; get port
2732 0000212B A808               <1>       test   al, RVRT ; wait for vertical retrace
2733 0000212D 74FB               <1>       jz     short n9 ; wait_display_enable
2734 0000212F B025               <1>       mov    al, 25h
2735 00002131 B2D8               <1>       mov    dl, 0D8h ; address control port
2736 00002133 EE                 <1>       out    dx, al ; turn off video during vertical retrace
2737 00002134 665A               <1>       pop    dx     ; #rows, #cols
2738 00002136 6658               <1>              pop    ax     ; offset
2739 00002138 6691               <1>       xchg   ax, cx ;
2740                              <1>       ; ecx = offset, al = line count, ah = attribute
2741                              <1>       ;
2742 0000213A 08C0               <1>       or     al, al
2743 0000213C C3                 <1>       retn
2744                              <1> n10:
2745                              <1>       ; Move rows
2746 0000213D 88D1               <1>       mov    cl, dl ; get # of cols to move
2747 0000213F 56                 <1>       push   esi
2748 00002140 57                 <1>       push   edi    ; save start address
2749                              <1> n10r:
2750 00002141 66A5               <1>       movsw         ; move that line on screen
2751 00002143 FEC9               <1>       dec    cl
2752 00002145 75FA               <1>         jnz     short n10r
2753 00002147 5F                 <1>       pop    edi
2754 00002148 5E                 <1>       pop    esi    ; recover addresses
2755 00002149 C3                 <1>       retn
2756                              <1> n11:
2757                              <1>       ; Clear rows
2758                              <1>                    ; dh =  #rows
2759 0000214A 88D1               <1>         mov  cl, dl ; get # of cols to clear
2760 0000214C 57                 <1>         push    edi     ; save address
2761                              <1> n11r:
2762 0000214D 66AB               <1>         stosw           ; store fill character
2763 0000214F FEC9               <1>       dec    cl
2764 00002151 75FA               <1>         jnz     short n11r
2765 00002153 5F                 <1>         pop     edi     ; recover address
2766 00002154 C3                 <1>       retn
2767                              <1>
2768                              <1> SCROLL_DOWN:
2769                              <1>       ; 07/07/2016
2770                              <1>       ; 27/06/2016
2771                              <1>       ; 26/06/2016
2772                              <1>       ; 12/05/2016
2773                              <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
2774                              <1>       ;
2775                              <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
2776                              <1>
2777                              <1> ;--------------------------------------------
2778                              <1> ; SCROLL DOWN
2779                              <1> ;     THIS ROUTINE MOVES THE CHARACTERS WITHIN A DEFINED
2780                              <1> ;     BLOCK DOWN ON THE SCREEN, FILLING THE TOP LINES
2781                              <1> ;     WITH A DEFINED CHARACTER
2782                              <1> ; INPUT
2783                              <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
00002155 38F5           <1>         cmp     ch, dh
                        <1>         ;ja     VIDEO_RETURN
00002157 7709           <1>         ja      short _s_d_retn ; 18/11/2020
00002159 38D1           <1>         cmp     cl, dl
                        <1>         ;ja     VIDEO_RETURN
0000215B 7705           <1>         ja      short _s_d_retn ; 18/11/2020
                        <1>         ;
0000215D E805000000     <1>         call    _scroll_down
                        <1> _s_d_retn:
00002162 E9F8F9FFFF     <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>         ; !!!!
00002167 FD             <1>         std           ; DIRECTION FOR SCROLL DOWN
                        <1>         ; !!!!
00002168 E879FFFFFF     <1>         call    test_line_count ; 16/01/2016
                        <1>
0000216D 8A25[DA6F0000] <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
00002173 80FC04         <1>         cmp     ah, 4
00002176 720A           <1>         jb      short _n0
00002178 80FC07         <1>         cmp     ah, 7 ; TEST FOR BW CARD
                        <1>                       ;       (80x25 text, mono)
0000217B 7405           <1>         je      short _n0 ; same with mode 3 for TRDOS 386
0000217D E903080000     <1>         jmp     GRAPHICS_DOWN
                        <1> _n0:
                        <1>         ; 07/07/2016
00002182 80FF07         <1>         cmp     bh, 7 ; video page number
00002185 7606           <1>         jna     short n12
00002187 8A3D[468A0100] <1>         mov     bh, [ACTIVE_PAGE]
                        <1>         ;
                        <1> n12:                ; CONTINUE_DOWN
0000218D 88DC           <1>         mov     ah, bl
0000218F 6650           <1>         push    ax      ; * ; save attribute in ah
00002191 6689D0         <1>         mov     ax, dx ; LOWER RIGHT CORNER
00002194 E860FFFFFF     <1>         call    scroll_position     ; GET REGEN LOCATION
00002199 741F           <1>         jz      short n14
0000219B 29CE           <1>         sub     esi, ecx  ; SI IS FROM ADDRESS
0000219D 88F5           <1>         mov     ch, dh  ; #rows in block
0000219F 28C5           <1>         sub     ch, al ; #rows to be moved
                        <1> n13:
000021A1 E897FFFFFF     <1>         call    n10     ; MOVE ONE ROW
                        <1>
000021A6 51             <1>         push    ecx
000021A7 8A0D[DC6F0000] <1>         mov     cl, [CRT_COLS]
000021AD 00C9           <1>         add     cl, cl
000021AF 29CE           <1>           sub   esi, ecx  ; next line
000021B1 29CF           <1>           sub   edi, ecx
000021B3 59             <1>           pop   ecx
                        <1>
000021B4 FECD           <1>         dec     ch      ; count of lines to move
000021B6 75E9           <1>         jnz     short n13 ; row loop
                        <1>         ; ch = 0
000021B8 88C6           <1>         mov     dh, al  ; #rows
                        <1> n14:
                        <1>         ; attribute in ah
000021BA B020           <1>         mov     al, ' '       ; fill with blanks
                        <1> n15:
000021BC E889FFFFFF     <1>         call    n11 ; 16/01/2016
                        <1>
000021C1 8A0D[DC6F0000] <1>         mov     cl, [CRT_COLS]
000021C7 00C9           <1>         add     cl, cl
000021C9 29CF           <1>           sub   edi, ecx
                        <1>
000021CB FECE           <1>         dec     dh
000021CD 75ED           <1>         jnz     short n15
                        <1>         ;
                        <1>         ; 18/11/2020
000021CF FC             <1>         cld     ; clear direction flag
                        <1>         ;
000021D0 E9FEFEFFFF     <1>         jmp     n16 ; 27/06/2016
                        <1>
                        <1> ;       cmp     bh, [ACTIVE_PAGE]
                        <1> ;       jne     short n16
                        <1> ;
                        <1> ;       ;cmp    byte [CRT_MODE], 7  ; is this the black and white card
                        <1> ;       ;je     short n16           ; if so, skip the mode reset
                        <1> ;
                        <1> ;       mov     al, [CRT_MODE_SET] ; get the value of mode set
                        <1> ;       mov     dx, 03D8h ; always set color card port
                        <1> ;       out     dx, al
```

```
                                <1> ;n16:
                                <1> ;       ; !!!!
                                <1> ;       cld             ; Clear direction flag !
                                <1> ;       ; !!!!
                                <1> ;       retn
                                <1>
                                <1> READ_AC_CURRENT:
                                <1>         ; 08/07/2016
                                <1>         ; 26/06/2016
                                <1>         ; 12/05/2016
                                <1>         ; 18/01/2016
                                <1>         ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ;
                                <1>         ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
                                <1>         ;
                                <1>         ; 08/07/2016
000021D5 803D[DA6F0000]07       <1>           cmp     byte [CRT_MODE], 7 ; 6!?
000021DC 7607                   <1>         jna     short read_ac_c
000021DE 31C0                   <1>         xor     eax, eax
000021E0 E97FF9FFFF             <1>           jmp     _video_return
                                <1> read_ac_c:
000021E5 E805000000             <1>         call    _read_ac_current
                                <1>         ; 12/05/2016
                                <1>           ;jmp     VIDEO_RETURN
000021EA E975F9FFFF             <1>         jmp     _video_return
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; READ_AC_CURRENT                                               :
                                <1> ;       THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER AT THE CURRENT     :
                                <1> ;       CURSOR POSITION AND RETURNS THEM TO THE CALLER                    :
                                <1> ; INPUT                                                                 :
                                <1> ;       (AH) = CURRENT CRT MODE                                         :
                                <1> ;       (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )                        :
                                <1> ;       (DS) = DATA SEGMENT                                             :
                                <1> ;       (ES) = REGEN SEGMENT                                                :
                                <1> ; OUTPUT                                                                :
                                <1> ;       (AL) = CHARACTER READ                                           :
                                <1> ;       (AH) = ATTRIBUTE READ                                           :
                                <1> ;--------------------------------------------------------------------------
                                <1>
                                <1> _read_ac_current:
                                <1>         ; 26/06/2016
                                <1>         ; 12/05/2016
                                <1>         ; 18/01/2016
                                <1>
000021EF 8A25[DA6F0000]         <1>         mov     ah, [CRT_MODE] ; current video mode
000021F5 80FC04                 <1>         cmp     ah, 4
000021F8 720A                   <1>         jb      short p10
                                <1>         ; 18/11/2020
                                <1>         ;cmp   byte [CRT_MODE], 4
                                <1>         ;jnb   GRAPHICS_READ ; 26/06/2016
                                <1>
000021FA 80FC07                 <1>         cmp     ah, 7 ; TEST FOR BW CARD (80x25 monochrome text)
000021FD 7405                   <1>         je      short p10       ; same with mode 3 in TRDOS 386
000021FF E9D7080000             <1>         jmp     GRAPHICS_READ
                                <1> p10:
00002204 E8F4FDFFFF             <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
00002209 803D[DA6F0000]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)
00002210 7516                   <1>         jne     short p13
                                <1>
                                <1>         ; 05/12/2020
00002212 3A3D[468A0100]         <1>         cmp      bh, [ACTIVE_PAGE]
00002218 750E                   <1>         jne     short p13
                                <1>
                                <1>         ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
                                <1> p11:
0000221A FB                     <1>         sti             ; enable interrupts first
                                <1>         ; 05/12/2020
0000221B 90                     <1>         nop
                                <1>         ; 18/11/2020
                                <1>         ;or    bl, bl
                                <1>         ;jnz   short p13 ; mode 01h (and mode 00h) - 40x25 color text
0000221C FA                     <1>         cli             ; block interrupts for single loop
0000221D EC                     <1>         in      al, dx ; get status from the adapter
0000221E A801                   <1>         test    al, RHRZ ; is horizontal retrace low
00002220 75F8                   <1>         jnz     short p11 ; wait until it is
                                <1> p12:                    ;  wait for either retrace high
00002222 EC                     <1>         in      al, dx ; get status again
00002223 A809                   <1>         test    al, RVRT+RHRZ ; is horizontal or vertical retrace high
00002225 74FB                   <1>         jz      short p12 ; wait until either retrace active
                                <1> ;p14:
00002227 FB                     <1>         sti
                                <1> p13:
00002228 81C600800B00           <1>         add     esi, 0B8000h
0000222E 668B06                 <1>         mov     ax, [esi]
                                <1>
```

```
2994 00002231 C3                <1>       retn   ; 18/01/2016
2995                            <1>
2996                            <1> WRITE_AC_CURRENT:
2997                            <1>       ; 08/07/2016
2998                            <1>       ; 26/06/2016
2999                            <1>       ; 24/06/2016
3000                            <1>       ; 12/05/2016
3001                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3002                            <1>       ;
3003                            <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
3004                            <1>       ;
3005                            <1> ;-----------------------------------------------------------------
3006                            <1> ; WRITE_AC_CURRENT                                :
3007                            <1> ;     THTS ROUTINE WRITES THE ATTRIBUTE AND CHARACTER            :
3008                            <1> ;     AT THE CURRENT CURSOR POSITION                      :
3009                            <1> ; INPUT                                           :
3010                            <1> ;     (AH) = CURRENT CRT MODE                     :
3011                            <1> ;     (BH) = DISPLAY PAGE                         :
3012                            <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE         :
3013                            <1> ;     (AL) = CHAR TO WRITE                        :
3014                            <1> ;     (BL) = ATTRIBUTE OF CHAR TO WRITE           :
3015                            <1> ;     (DS) = DATA SEGMENT                         :
3016                            <1> ;     (ES) = REGEN SEGMENT                        :
3017                            <1> ; OUTPUT                                          :
3018                            <1> ;     DISPLAY REGEN BUFFER UPDATED                :
3019                            <1> ;-----------------------------------------------------------------
3020                            <1>
3021                            <1>       ; 08/07/2016
3022 00002232 803D[DA6F0000]07  <1>       cmp    byte [CRT_MODE], 7 ; 6!?
3023 00002239 760A              <1>       jna    short write_ac_c
3024                            <1>
3025 0000223B E8110B0000        <1>       call   vga_write_char_attr
3026 00002240 E91AF9FFFF        <1>       jmp     VIDEO_RETURN
3027                            <1>
3028                            <1> write_ac_c:
3029 00002245 E834000000        <1>       call   _write_c_current
3030                            <1>
3031 0000224A 0FB6F7            <1>       movzx  esi, bh ; video page number (0 to 7)
3032 0000224D 889E[E36F0000]    <1>       mov    [esi+chr_attrib], bl ; color/attribute
3033                            <1>
3034 00002253 E907F9FFFF        <1>         jmp     VIDEO_RETURN
3035                            <1>
3036                            <1> WRITE_C_CURRENT:
3037                            <1>       ; 08/07/2016
3038                            <1>       ; 26/06/2016
3039                            <1>       ; 12/05/2016
3040                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3041                            <1>       ;
3042                            <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
3043                            <1>       ;
3044                            <1> ;-----------------------------------------------------------------
3045                            <1> ; WRITE_C_CURRENT                                 :
3046                            <1> ;     THIS ROUTINE WRITES THE CHARACTER AT                :
3047                            <1> ;     THE CURRENT CURSOR POSITION, ATTRIBUTE UNCHANGED    :
3048                            <1> ; INPUT                                           :
3049                            <1> ;     (AH) = CURRENT CRT MODE                     :
3050                            <1> ;     (BH) = DISPLAY PAGE                         :
3051                            <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE         :
3052                            <1> ;     (AL) = CHAR TO WRITE                        :
3053                            <1> ;     (DS) = DATA SEGMENT                         :
3054                            <1> ;     (ES) = REGEN SEGMENT                        :
3055                            <1> ; OUTPUT                                          :
3056                            <1> ;     DISPLAY REGEN BUFFER UPDATED                :
3057                            <1> ;-----------------------------------------------------------------
3058                            <1>
3059                            <1>       ; 08/07/2016
3060 00002258 803D[DA6F0000]07  <1>       cmp    byte [CRT_MODE], 7 ; 6!?
3061 0000225F 760A              <1>       jna    short write_c_c
3062                            <1>
3063 00002261 E8EB0A0000        <1>       call   vga_write_char_only
3064 00002266 E9F4F8FFFF        <1>       jmp     VIDEO_RETURN
3065                            <1>
3066                            <1> write_c_c:
3067                            <1>       ;and   bh, 7 ; video page number (<= 7)
3068 0000226B 0FB6F7            <1>       movzx  esi, bh
3069 0000226E 8A9E[E36F0000]    <1>       mov    bl, [esi+chr_attrib]
3070                            <1>
3071 00002274 E805000000        <1>       call   _write_c_current
3072 00002279 E9E1F8FFFF        <1>         jmp     VIDEO_RETURN
3073                            <1>
3074                            <1> _write_c_current:  ; from 'write_tty'
3075                            <1>       ; 18/11/2020
3076                            <1>       ; 26/06/2016
3077                            <1>       ; 24/06/2016
3078                            <1>       ; 12/05/2016
3079                            <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3080                            <1>       ; 30/08/2014 (Retro UNIX 386 v1)
3081                            <1>       ; 18/01/2014
3082                            <1>       ; 04/12/2013
3083                            <1>       ;
3084                            <1>       ; VIDEO.ASM - 06/10/85 VIDEO DISPLAY BIOS
3085                            <1>
3086                            <1>       ; 18/11/2020
3087 0000227E 8A25[DA6F0000]    <1>       mov    ah, [CRT_MODE] ; current video mode
3088 00002284 80FC04            <1>       cmp    ah, 4
3089 00002287 720A              <1>       jb     short p40
3090                            <1>
3091                            <1>       ;cmp   byte [CRT_MODE], 4
3092                            <1>         ;jnb GRAPHICS_WRITE ; 26/06/2016
3093                            <1>
3094 00002289 80FC07            <1>       cmp    ah, 7 ; TEST FOR BW CARD
3095 0000228C 7405              <1>       je     short p40
3096 0000228E E998070000        <1>       jmp    GRAPHICS_WRITE
3097                            <1> p40:
3098                            <1>       ; al = character
```

```
3099                                    <1>       ; bl = color/attribute
3100                                    <1>       ; bh = video page
3101                                    <1>       ; cx = count of characters to write
3102 00002293 6652                     <1>       push  dx
3103 00002295 88DC                     <1>       mov   ah, bl  ; color/attribute (12/05/2016)
3104 00002297 6650                     <1>       push  ax    ; save character & attribute/color
3105 00002299 6651                     <1>       push  cx
3106 0000229B E85DFDFFFF               <1>       call  find_position  ; get regen location and port address
3107 000022A0 6659                     <1>       pop   cx
3108                                    <1>       ; esi = regen location
3109                                    <1>       ; dx = status port
3110                                    <1>       ;
3111 000022A2 81C600800B00             <1>       add   esi, 0B8000h ; 30/08/2014 (crt_base)
3112                                    <1>       ;
3113                                    <1>       ; 18/11/2020
3114                                    <1>       ; convert display mode to a zero value
3115                                    <1>       ; for 80 column color mode
3116                                    <1>       ;mov   ah, [CRT_MODE]
3117                                    <1>       ;sub   ah, 2
3118                                    <1>       ;shr   ah, 1
3119                                    <1>       ;jnz   short p44    ; 26/06/2016
3120                                    <1>
3121                                    <1>       ; 05/12/2020
3122                                    <1>       ; 18/11/2020 (TRDOS 386 v2.0.3)
3123                                    <1>       ;xor   bl, bl ; 0
3124 000022A8 803D[DA6F0000]03         <1>       cmp   byte [CRT_MODE], 03h ; 80x25 color text
3125                                    <1>       ;je    short p41    ; Note: Only mode 03h and mode 01h are
3126                                    <1>       ;inc   bl    ; 1  ;       in use by TRDOS 386 as text modes
3127                                    <1>       ;jmp   short p43    ;       (07h, 00h and 02h are redirected)
3128                                    <1>       ; 05/12/2020
3129 000022AF 751A                     <1>       jne   short p44
3130                                    <1> p46:
3131                                    <1>       ; 05/12/2020
3132 000022B1 3A3D[468A0100]           <1>       cmp    bh, [ACTIVE_PAGE]
3133 000022B7 7512                     <1>       jne   short p44
3134                                    <1>
3135                                    <1>       ; WAIT FOR HORIZONTAL RETRACE OR VERTICAL RETRACE IF COLOR 80
3136                                    <1> p41:
3137                                    <1>       ; 05/12/2020
3138 000022B9 FB                       <1>       sti          ; enable interrupts first
3139 000022BA 90                       <1>       nop
3140                                    <1>       ; 18/11/2020
3141                                    <1>       ;or   bl, bl
3142                                    <1>       ;jnz  short p44 ; mode 01h (and mode 00h) - 40x25 color text
3143 000022BB FA                       <1>       cli          ; block interrupts for single loop
3144 000022BC EC                       <1>       in    al, dx ; get status from the adapter
3145 000022BD A808                     <1>       test  al, RVRT ; check for vertical retrace first
3146 000022BF 7509                     <1>       jnz   short p43 ; Do fast write now if vertical retrace
3147 000022C1 A801                     <1>       test  al, RHRZ ; is horizontal retrace low
3148 000022C3 75F4                     <1>       jnz   short p41 ; wait until it is
3149                                    <1> p42:             ;  wait for either retrace high
3150 000022C5 EC                       <1>       in    al, dx ; get status again
3151 000022C6 A809                     <1>       test  al, RVRT+RHRZ ; is horizontal or vertical retrace high
3152 000022C8 74FB                     <1>       jz    short p42 ; wait until either retrace active
3153                                    <1> p43:
3154 000022CA FB                       <1>       sti
3155                                    <1> p44:
3156 000022CB 668B0424                 <1>       mov   ax, [esp] ; restore the character (al) & attribute (ah)
3157 000022CF 668906                   <1>       mov   [esi], ax
3158                                    <1>
3159 000022D2 6649                     <1>       dec   cx
3160 000022D4 740D                     <1>       jz    short p45
3161                                    <1>
3162 000022D6 46                       <1>       inc   esi
3163 000022D7 46                       <1>       inc   esi
3164                                    <1>
3165                                    <1>       ; 05/12/2020
3166 000022D8 803D[DA6F0000]03         <1>       cmp   byte [CRT_MODE], 03h
3167 000022DF 75EA                     <1>       jne   short p44
3168                                    <1>       ;jmp  short p41
3169 000022E1 EBCE                     <1>       jmp   short p46
3170                                    <1> p45:
3171 000022E3 6658                     <1>       pop   ax
3172 000022E5 665A                     <1>       pop   dx
3173 000022E7 C3                       <1>       retn
3174                                    <1>
3175                                    <1> ; 09/07/2016
3176                                    <1> ; 26/06/2016
3177                                    <1> ; 24/06/2016
3178                                    <1> ; 12/05/2016
3179                                    <1> ; 18/01/2016
3180                                    <1> ; 16/01/2016 - TRDOS 386 (TRDOS v2.0)
3181                                    <1> ; 30/06/2015
3182                                    <1> ; 27/06/2015
3183                                    <1> ; 11/03/2015
3184                                    <1> ; 02/09/2014
3185                                    <1> ; 30/08/2014
3186                                    <1> ; VIDEO FUNCTIONS
3187                                    <1> ; (write_tty - Retro UNIX 8086 v1 - U9.ASM, 01/02/2014)
3188                                    <1>
3189                                    <1> WRITE_TTY:
3190                                    <1>       ; 09/12/2017
3191                                    <1>       ; 09/07/2016
3192                                    <1>       ; 01/07/2016
3193                                    <1>       ; 26/06/2016
3194                                    <1>       ; 24/06/2016
3195                                    <1>       ; 13/05/2016
3196                                    <1>       ; 12/05/2016
3197                                    <1>       ; 30/01/2016
3198                                    <1>       ; 18/01/2016
3199                                    <1>       ; 16/01/2016 (TRDOS 386 = TRDOS v2.0)
3200                                    <1>       ; 13/08/2015
3201                                    <1>       ; 02/09/2014
3202                                    <1>       ; 30/08/2014 (Retro UNIX 386 v1 - beginning)
3203                                    <1>       ; 01/02/2014 (Retro UNIX 8086 v1 - last update)
```

```
                            <1>       ; 03/12/2013 (Retro UNIX 8086 v1 - beginning)
                            <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
                            <1>       ;
                            <1>       ; INPUT -> AL = Character to be written
                            <1>       ;         BL = Color (Forecolor, Backcolor)
                            <1>       ;         BH = Video Page (0 to 7)
                            <1>
                            <1>       ; 09/07/2016
000022E8 803D[DA6F0000]07   <1>         cmp     byte [CRT_MODE], 7
000022EF 760A               <1>       jna    short write_tty_cga
                            <1>
000022F1 E8460D0000         <1>       call   vga_write_teletype
000022F6 E964F8FFFF         <1>       jmp    VIDEO_RETURN
                            <1>
                            <1> write_tty_cga:
                            <1>       ; 13/05/2016
                            <1>       ;call  _write_tty
                            <1>       ; 01/07/2016
000022FB E818000000         <1>       call   _write_tty_m3
00002300 E95AF8FFFF         <1>       jmp    VIDEO_RETURN
                            <1>
                            <1> RVRT  equ    00001000b    ; VIDEO VERTICAL RETRACE BIT
                            <1> RHRZ  equ    00000001b    ; VIDEO HORIZONTAL RETRACE BIT
                            <1>
                            <1> ; Derived from "WRITE_TTY" procedure of IBM "pc-at" rombios source code
                            <1> ; (06/10/1985), 'video.asm', INT 10H, VIDEO_IO
                            <1> ;
                            <1> ; 06/10/85  VIDEO DISPLAY BIOS
                            <1> ;
                            <1> ;--- WRITE_TTY ----------------------------------------------------------------
                            <1> ;                                                                           :
                            <1> ;   THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE                :
                            <1> ;   VIDEO CARDS. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT              :
                            <1> ;   CURSOR POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION.          :
                            <1> ;   IF THE CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN           :
                            <1> ;   IS SET TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW            :
                            <1> ;   ROW VALUE LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW,       :
                            <1> ;   FIRST COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE.            :
                            <1> ;   WHEN THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE           :
                            <1> ;   NEWLY BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS     :
                            <1> ;   LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE,            :
                            <1> ;   THE 0 COLOR IS USED.                                                   :
                            <1> ;   ENTRY --                                                               :
                            <1> ;     (AH) = CURRENT CRT MODE                                              :
                            <1> ;     (AL) = CHARACTER TO BE WRITTEN                                       :
                            <1> ;         NOTE THAT BACK SPACE, CARRIAGE RETURN, BELL AND LINE FEED ARE    :
                            <1> ;         HANDLED AS COMMANDS RATHER THAN AS DISPLAY GRAPHICS CHARACTERS   :
                            <1> ;     (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A GRAPHICS MODE  :
                            <1> ;   EXIT --                                                                :
                            <1> ;     ALL REGISTERS SAVED                                                  :
                            <1> ;-------------------------------------------------------------------------------
                            <1>
                            <1> ; 18/11/2020 (TRDOS 386 v2.0.3)
                            <1> ; 09/12/2017
                            <1> ; 08/07/2016
                            <1> ; 26/06/2016
                            <1> ; 24/06/2016
                            <1> _write_tty:
                            <1>       ; 13/05/2016
                            <1>       ; --- 18/11/2020 ---
                            <1>       ; NOTE:
                            <1>       ; Only kernel calls "_write_tty" procedure...
                            <1>       ; TRDOS 386 v2 kernel uses video mode 3 for displaying
                            <1>       ; (some error) messages and also mainprog command interpreter
                            <1>       ; (in kernel) uses "_write_tty".
                            <1>       ; So, here video mode must be set to 3 if it is not 3.
                            <1>
00002305 FA                 <1>       cli    ; disable interrupts
                            <1>       ;
                            <1>       ; 01/09/2014
00002306 803D[DA6F0000]03   <1>       cmp    byte [CRT_MODE], 3
0000230D 7409               <1>       je     short _write_tty_m3
                            <1>       ;
                            <1> set_mode_3:
0000230F 53                 <1>       push   ebx
00002310 50                 <1>       push   eax
                            <1>       ;call  _set_mode
                            <1>       ; 18/11/2020
00002311 E858F8FFFF         <1>       call   set_txt_mode  ; set video mode to 03h
00002316 58                 <1>       pop    eax
00002317 5B                 <1>       pop    ebx
                            <1>       ;
                            <1> _write_tty_m3: ; 24/06/2016 (m3 -> _write_tty_m3)
00002318 0FB6F7             <1>       movzx  esi, bh ; 12/05/2016
0000231B 66D1E6             <1>       shl    si, 1
0000231E 81C6[368A0100]     <1>       add    esi, CURSOR_POSN
00002324 668B16             <1>       mov    dx, [esi]
                            <1>       ;
                            <1>       ; dx now has the current cursor position
                            <1>       ;
00002327 3C0D               <1>       cmp    al, 0Dh      ; CR   ; is it carriage return or control character
00002329 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>       ;
0000232B 66B90100           <1>       mov    cx, 1  ; 24/06/2016
                            <1>       ; cx = count of characters to write
                            <1>       ;
0000232F E84AFFFFFF         <1>       call   _write_c_current ; 16/01/2015
                            <1>       ;
                            <1>       ; position the cursor for next char
```

```
3309 00002334 FEC2                <1>       inc    dl           ; next column
3310 00002336 3A15[DC6F0000]      <1>       cmp    dl, [CRT_COLS]  ; test for column overflow
3311 0000233C 7561                <1>         jne     _set_cpos
3312 0000233E B200                <1>       mov    dl, 0        ; column = 0
3313                              <1> u10:                      ; (line feed found)
3314 00002340 80FE18              <1>       cmp    dh, 25-1     ; check for last row
3315 00002343 7220                <1>       jb     short u6
3316                              <1>       ;
3317                              <1>       ; scroll required
3318                              <1> u1:
3319                              <1>       ; SET CURSOR POSITION (04/12/2013)
3320 00002345 E855000000          <1>       call   _set_cpos
3321                              <1>       ;
3322                              <1>       ; determine value to fill with during scroll
3323                              <1> u2:
3324                              <1>       ; bh = video page number
3325                              <1>       ;
3326 0000234A E8A0FEFFFF          <1>       call   _read_ac_current ; 18/01/2016
3327                              <1>       ;
3328                              <1>       ; al = character, ah = attribute
3329                              <1>       ; bh = video page number
3330                              <1>       ; 18/11/2020
3331 0000234F 88E3                <1>       mov    bl, ah ; color/attribute
3332                              <1> u3:
3333                              <1>       ;;mov  ax, 0601h    ; scroll one line
3334                              <1>       ;;sub  cx, cx       ; upper left corner
3335                              <1>       ;;mov  dh, 25-1     ; lower right row
3336                              <1>       ;;;mov dl, [CRT_COLS]
3337                              <1>       ;mov   dl, 80       ; lower right column
3338                              <1>       ;;dec  dl
3339                              <1>       ;;mov  dl, 79
3340                              <1>
3341                              <1>       ;;call scroll_up    ; 04/12/2013
3342                              <1>       ;;; 11/03/2015
3343                              <1>       ; 02/09/2014
3344                              <1>       ;;;mov cx, [crt_ulc] ; Upper left corner  (0000h)
3345                              <1>       ;;;mov dx, [crt_lrc] ; Lower right corner (184Fh)
3346                              <1>       ; 11/03/2015
3347 00002351 6629C9              <1>       sub    cx, cx
3348                              <1>       ;mov   dx, 184Fh ; dl = 79 (column), dh = 24 (row)
3349                              <1>       ; 18/11/2020
3350 00002354 B618                <1>       mov    dh, 25-1
3351 00002356 8A15[DC6F0000]      <1>       mov    dl, [CRT_COLS]
3352 0000235C FECA                <1>       dec    dl
3353                              <1>       ;
3354 0000235E B001                <1>       mov    al, 1        ; scroll 1 line up
3355                              <1>              ; ah = attribute
3356                              <1>       ;mov   bl, al ; 12/05/2016
3357 00002360 E9E0FCFFFF          <1>       jmp    _scroll_up   ; 16/01/2016
3358                              <1> ;u4:
3359                              <1>       ;;int  10h          ; video-call return
3360                              <1>                           ; scroll up the screen
3361                              <1>                           ; tty return
3362                              <1> ;u5:
3363                              <1>       ;retn               ; return to the caller
3364                              <1>
3365                              <1> u6:                       ; set-cursor-inc
3366 00002365 FEC6                <1>       inc    dh           ; next row
3367                              <1>                           ; set cursor
3368                              <1> ;u7:
3369                              <1>       ;;mov  ah, 02h
3370                              <1>       ;;jmp  short u4     ; establish the new cursor
3371                              <1>       ;call  _set_cpos
3372                              <1>       ;jmp   short u5
3373 00002367 EB36                <1>       jmp     _set_cpos
3374                              <1>
3375                              <1>       ; check for control characters
3376                              <1> u8:
3377 00002369 7432                <1>       je     short u9
3378 0000236B 3C0A                <1>       cmp    al, 0Ah            ; is it a line feed (0Ah)
3379 0000236D 74D1                <1>       je     short u10
3380 0000236F 3C07                <1>       cmp    al, 07h      ; is it a bell
3381 00002371 747E                <1>       je     short u11
3382 00002373 3C08                <1>       cmp    al, 08h            ; is it a backspace
3383                              <1>       ;jne   short u0
3384 00002375 741E                <1>       je     short bs     ; 12/12/2013
3385                              <1>       ; 12/12/2013 (tab stop)
3386 00002377 3C09                <1>       cmp    al, 09h            ; is it a tab stop
3387 00002379 75B0                <1>       jne    short u0
3388 0000237B 88D0                <1>       mov    al, dl
3389                              <1>       ;cbw
3390 0000237D 30E4                <1>       xor    ah, ah ; 09/12/2017
3391 0000237F B108                <1>       mov    cl, 8
3392 00002381 F6F1                <1>       div    cl
3393 00002383 28E1                <1>       sub    cl, ah
3394                              <1> ts:
3395                              <1>       ; 02/09/2014
3396                              <1>       ; 01/09/2014
3397                              <1>       ;mov   al, 20h
3398                              <1> tsloop:
3399 00002385 6651                <1>       push   cx
3400                              <1>       ; 18/11/2020
3401                              <1>       ;push  ax
3402                              <1>       ;;mov  bh, [ACTIVE_PAGE]
3403                              <1>       ; 05/12/2020
3404                              <1>       ;push  bx
3405 00002387 B020                <1>       mov    al, 20h ; al = space (blank)
3406                              <1>                      ; bl = color/attribute
3407 00002389 E88AFFFFFF          <1>       call   _write_tty_m3 ; 24/06/2016 (m3 -> _write_tty_m3)
3408                              <1>       ; 05/12/2020
3409                              <1>       ; bx is preserved in '_write_tty_m3'
3410                              <1>       ; 18/11/2020
3411                              <1>       ;pop   bx  ; BL = color/attribute, Bh = video page
3412                              <1>       ;pop   ax  ; AL = character
3413 0000238E 6659                <1>       pop    cx
```

```
3414 00002390 FEC9              <1>       dec    cl
3415 00002392 75F1              <1>       jnz    short tsloop
3416 00002394 C3                <1>       retn
3417                            <1> bs:
3418                            <1>       ; back space found
3419                            <1>
3420 00002395 08D2              <1>       or     dl, dl              ; is it already at start of line
3421                            <1>       ;je    short u7     ; set_cursor
3422 00002397 7406              <1>       jz     short _set_cpos
3423 00002399 664A              <1>       dec    dx                  ; no -- just move it back
3424                            <1>       ;jmp   short u7
3425 0000239B EB02              <1>       jmp    short _set_cpos
3426                            <1>
3427                            <1>       ; carriage return found
3428                            <1> u9:
3429 0000239D B200              <1>       mov    dl, 0        ; move to first column
3430                            <1>       ;jmp   short u7
3431                            <1>       ;jmp   short _set_cpos ; 30/01/2016
3432                            <1>
3433                            <1>       ; line feed found
3434                            <1> ;u10:
3435                            <1> ;     cmp    dh, 25-1     ; bottom of screen
3436                            <1> ;     jne    short u6     ; no, just set the cursor
3437                            <1> ;       jmp     u1              ; yes, scroll the screen
3438                            <1>
3439                            <1> _set_cpos:
3440                            <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
3441                            <1>       ; 27/06/2015
3442                            <1>       ; 01/09/2014
3443                            <1>       ; 30/08/2014 (Retro UNIX 386 v1)
3444                            <1>       ;
3445                            <1>       ; 04/12/2013 - 12/12/2013 (Retro UNIX 8086 v1)
3446                            <1>       ;
3447                            <1>       ; VIDEO.ASM - 06/10/85  VIDEO DISPLAY BIOS
3448                            <1>       ;
3449                            <1> ;--------------------------------------------
3450                            <1> ; SET_CPOS
3451                            <1> ;     THIS ROUTINE SETS THE CURRENT CURSOR POSITION TO THE
3452                            <1> ;     NEW X-Y VALUES PASSED
3453                            <1> ; INPUT
3454                            <1> ;     DX - ROW,COLUMN OF NEW CURSOR
3455                            <1> ;     BH - DISPLAY PAGE OF CURSOR
3456                            <1> ; OUTPUT
3457                            <1> ;     CURSOR ID SET AT 6845 IF DISPLAY PAGE IS CURRENT DISPLAY
3458                            <1> ;--------------------------------------------
3459                            <1>       ;
3460 0000239F BE[368A0100]      <1>       mov    esi, CURSOR_POSN
3461 000023A4 0FB6C7            <1>         movzx   eax, bh   ; BH = video page number
3462                            <1> ;     or     al, al
3463                            <1> ;     jz     short _set_cpos_0
3464 000023A7 D0E0              <1>         shl     al, 1   ; word offset
3465 000023A9 01C6              <1>         add     esi, eax
3466                            <1> ;_set_cpos_0:
3467 000023AB 668916            <1>       mov    [esi], dx ; save the pointer
3468 000023AE 383D[468A0100]    <1>       cmp    [ACTIVE_PAGE], bh
3469 000023B4 7532              <1>       jne    short m17
3470                            <1>       ;call  m18    ; CURSOR SET
3471                            <1> ;m17:              ; SET_CPOS_RETURN
3472                            <1>       ; 01/09/2014
3473                            <1> ;     retn
3474                            <1>             ; DX  = row/column
3475                            <1> m18:
3476 000023B6 E830FCFFFF        <1>       call   position ; determine location in regen buffer
3477 000023BB 668B0D[348A0100]  <1>       mov    cx, [CRT_START]
3478 000023C2 6601C1            <1>       add    cx, ax  ; add char position in regen buffer
3479                            <1>                    ; to the start address (offset) for this page
3480 000023C5 66D1E9            <1>       shr    cx, 1  ; divide by 2 for char only count
3481 000023C8 B40E              <1>       mov    ah, 14 ; register number for cursor
3482                            <1>       ;call  m16    ; output value to the 6845
3483                            <1>       ;retn
3484                            <1>
3485                            <1>       ;----- THIS ROUTINE OUTPUTS THE CX REGISTER
3486                            <1>       ;     TO THE 6845 REGISTERS NAMED IN (AH)
3487                            <1> m16:
3488 000023CA FA                <1>       cli
3489                            <1>       ;mov   dx, [addr_6845] ; address register
3490 000023CB 66BAD403          <1>       mov    dx, 03D4h ; I/O address of color card
3491 000023CF 88E0              <1>       mov    al, ah ; get value
3492 000023D1 EE                <1>       out    dx, al ; register set
3493 000023D2 6642              <1>       inc    dx     ; data register
3494 000023D4 EB00              <1>       jmp    $+2    ; i/o delay
3495 000023D6 88E8              <1>       mov    al, ch ; data
3496 000023D8 EE                <1>       out    dx, al
3497 000023D9 664A              <1>       dec    dx
3498 000023DB 88E0              <1>       mov    al, ah
3499 000023DD FEC0              <1>       inc    al     ; point to other data register
3500 000023DF EE                <1>       out    dx, al ; set for second register
3501 000023E0 6642              <1>       inc    dx
3502 000023E2 EB00              <1>       jmp    $+2    ; i/o delay
3503 000023E4 88C8              <1>       mov    al, cl ; second data value
3504 000023E6 EE                <1>       out    dx, al
3505 000023E7 FB                <1>       sti
3506                            <1> m17:
3507 000023E8 C3                <1>       retn
3508                            <1>
3509                            <1> _beep:
3510                            <1>       ; 12/02/2021 (TRDOS v2.0.3)
3511 000023E9 FA                <1>       cli
3512 000023EA E811000000        <1>       call   beep
3513 000023EF FB                <1>       sti
3514 000023F0 C3                <1>       retn
3515                            <1>
3516                            <1> beeper:
3517                            <1>       ; 04/08/2016
3518                            <1>       ; 12/05/2016 - TRDOS 386 (TRDOS v2.0)
```

```
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <1>       ; 18/01/2014
                                <1>       ; 03/12/2013
                                <1>       ; bell found
                                <1> u11:
000023F1 FB                     <1>       sti
000023F2 3A3D[468A0100]         <1>       cmp    bh, [ACTIVE_PAGE]
000023F8 7553                   <1>       jne    short u12     ; Do not sound the beep
                                <1>                            ; if it is not written on the active page
                                <1> beeper_gfx: ; 04/08/2016
000023FA 66B93305               <1>       mov    cx, 1331      ; divisor for 896 hz tone
000023FE B31F                   <1>       mov    bl, 31        ; set count for 31/64 second for beep
                                <1>       ;call  beep          ; sound the pod bell
                                <1>       ;jmp   short u5      ; tty_return
                                <1>       ;retn
                                <1>
                                <1> TIMER equ    040h                ; 8254 TIMER - BASE ADDRESS
                                <1> PORT_B       equ    061h         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
                                <1> GATE2 equ    00000001b     ; TIMER 2 INPUT CATE CLOCK BIT
                                <1> SPK2  equ    00000010b     ; SPEAKER OUTPUT DATA ENABLE BIT
                                <1>
                                <1> beep:
                                <1>       ; 12/02/2021
                                <1>       ; 07/02/2015
                                <1>       ; 30/08/2014 (Retro UNIX 386 v1)
                                <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>
00002400 9C                     <1>       pushfd  ; 18/01/2014      ; save interrupt status
00002401 FA                     <1>       cli                ; block interrupts during update
00002402 B0B6                   <1>       mov    al, 10110110b; select timer 2, lsb, msb binary
00002404 E643                   <1>       out    TIMER+3, al  ; write timer mode register
00002406 EB00                   <1>       jmp    $+2          ; I/O delay
00002408 88C8                   <1>       mov    al, cl       ; divisor for hz (low)
0000240A E642                   <1>       out    TIMER+2,AL   ; write timer 2 count - lsb
0000240C EB00                   <1>       jmp    $+2          ; I/O delay
0000240E 88E8                   <1>       mov    al, ch       ; divisor for hz (high)
00002410 E642                   <1>       out    TIMER+2, al  ; write timer 2 count - msb
00002412 E461                   <1>       in     al, PORT_B   ; get current setting of port
00002414 88C4                   <1>       mov    ah, al       ; save that setting
00002416 0C03                   <1>       or     al, GATE2+SPK2      ; gate timer 2 and turn speaker on
00002418 E661                   <1>       out    PORT_B, al   ; and restore interrupt status
                                <1>       ; 12/02/2021
0000241A 9D                     <1>       popfd  ; 18/01/2014
                                <1>       ;sti
                                <1> g7:                      ; 1/64 second per count (bl)
0000241B B90B040000             <1>       mov    ecx, 1035    ; delay count for 1/64 of a second
00002420 E829000000             <1>       call   waitf        ; go to beep delay 1/64 count
00002425 FECB                   <1>       dec    bl           ; (bl) length count expired?
00002427 75F2                   <1>       jnz    short g7     ; no - continue beeping speaker
                                <1>       ;
00002429 9C                     <1>       pushfd ; 12/02/2021 ; save interrupt status
0000242A FA                     <1>       cli    ; 18/01/2014 ; block interrupts during update
0000242B E461                   <1>       in     al, PORT_B   ; get current port value
                                <1>         ;or      al, not (GATE2+SPK2) ; isolate current speaker bits in case
0000242D 0CFC                   <1>         or      al, ~(GATE2+SPK2)
0000242F 20C4                   <1>         and ah, al       ; someone turned them off during beep
00002431 88E0                   <1>       mov    al, ah       ; recover value of port
                                <1>         ;or      al, not (GATE2+SPK2) ; force speaker data off
00002433 0CFC                   <1>       or     al, ~(GATE2+SPK2) ; isolate current speaker bits in case
00002435 E661                   <1>       out    PORT_B, al   ; and stop speaker timer
00002437 9D                     <1>       popfd  ; 12/02/2021       ; restore interrupt flag state
                                <1>       ;sti
00002438 B90B040000             <1>       mov    ecx, 1035    ; force 1/64 second delay (short)
0000243D E80C000000             <1>       call   waitf        ; minimum delay between all beeps
00002442 9C                     <1>       pushfd              ; save interrupt status
00002443 FA                     <1>       cli                 ; block interrupts during update
00002444 E461                   <1>       in     al, PORT_B   ; get current port value in case
00002446 2403                   <1>       and    al, GATE2+SPK2      ; someone turned them on
00002448 08E0                   <1>       or     al, ah       ; recover value of port_b
0000244A E661                   <1>       out    PORT_B, al   ; restore speaker status
0000244C 9D                     <1>       popfd               ; restore interrupt flag state
                                <1> u12:
0000244D 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
```

```
3624                                    <1>
3625                                    <1> ; 06/02/2015 (unix386.s <-- dsectrm2.s)
3626                                    <1> ; 17/12/2014 (dsectrm2.s)
3627                                    <1> ; WAITF
3628                                    <1> ; /// IBM PC-XT Model 286 System BIOS Source Code - Test 4 - 06/10/85 ///
3629                                    <1> ;
3630                                    <1> ;---WAITF-----------------------------------------------------------------------
3631                                    <1> ;     FIXED TIME WAIT ROUTINE (HARDWARE CONTROLLED - NOT PROCESSOR)
3632                                    <1> ; ENTRY:
3633                                    <1> ;     (CX) = COUNT OF 15.085737 MICROSECOND INTERVALS TO WAIT
3634                                    <1> ;            MEMORY REFRESH TIMER 1 OUTPUT USED AS REFERENCE
3635                                    <1> ; EXIT:
3636                                    <1> ;                   AFTER (CX) TIME COUNT (PLUS OR MINUS 16 MICROSECONDS)
3637                                    <1> ;     (CX) = 0
3638                                    <1> ;-----------------------------------------------------------------------------
3639                                    <1>
3640                                    <1> ; Refresh period: 30 micro seconds (15-80 us)
3641                                    <1> ; (16/12/2014 - AWARDBIOS 1999 - ATORGS.ASM, WAIT_REFRESH)
3642                                    <1>
3643                                    <1> ;WAITF:                             ; DELAY FOR (CX)*15.085737 US
3644 0000244E 6650                     <1>       PUSH   AX                   ; SAVE WORK REGISTER (AH)
3645                                    <1>       ; 16/12/2014
3646                                    <1>       ;shr   cx, 1                ; convert to count of 30 micro seconds
3647 00002450 D1E9                     <1>       shr    ecx, 1 ; 21/02/2015
3648                                    <1> ;17/12/2014
3649                                    <1> ;WAITF1:
3650                                    <1> ;     IN     AL, PORT_B   ;061h  ; READ CURRENT COUNTER OUTPUT STATUS
3651                                    <1> ;     AND    AL, REFRESH_BIT     ;00010000b ; MASK FOR REFRESH DETERMINE BIT
3652                                    <1> ;     CMP    AL, AH              ; DID IT JUST CHANGE
3653                                    <1> ;     JE     short WAITF1        ; WAIT FOR A CHANGE IN OUTPUT LINE
3654                                    <1> ;     MOV    AH, AL              ; SAVE NEW FLAG STATE
3655                                    <1> ;     LOOP   WAITF1              ; DECREMENT HALF CYCLES TILL COUNT END
3656                                    <1>       ;
3657                                    <1>       ; 17/12/2014
3658                                    <1>       ;
3659                                    <1>       ; Modification from 'WAIT_REFRESH' procedure of AWARD BIOS - 1999
3660                                    <1>       ;
3661                                    <1> ;WAIT_REFRESH:  Uses port 61, bit 4 to have CPU speed independent waiting.
3662                                    <1> ;     INPUT:  CX = number of refresh periods to wait
3663                                    <1> ;                  (refresh periods = 1 per 30 microseconds on most machines)
3664                                    <1> WR_STATE_0:
3665 00002452 E461                     <1>       IN     AL,PORT_B           ; IN AL,SYS1
3666 00002454 A810                     <1>       TEST   AL,010H
3667 00002456 74FA                     <1>       JZ     SHORT WR_STATE_0
3668                                    <1> WR_STATE_1:
3669 00002458 E461                     <1>       IN     AL,PORT_B           ; IN AL,SYS1
3670 0000245A A810                     <1>       TEST   AL,010H
3671 0000245C 75FA                     <1>       JNZ    SHORT WR_STATE_1
3672 0000245E E2F2                     <1>         LOOP    WR_STATE_0
3673                                    <1>       ;
3674 00002460 6658                     <1>       POP    AX                  ; RESTORE (AH)
3675 00002462 C3                       <1>       RETn                       ; (CX) = 0
3676                                    <1>
3677                                    <1> ; 09/07/2016
3678                                    <1> ; 01/07/2016
3679                                    <1> ; 24/06/2016
3680                                    <1> ; 23/06/2016 - TRDOS 386 (TRDOS v2.0)
3681                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
3682                                    <1> ;-----------------------------------------------------------------------------
3683                                    <1> ; WRITE_STRING                                                          :
3684                                    <1> ;     THIS ROUTINE WRITES A STRING OF CHARACTERS TO THE CRT.                    :
3685                                    <1> ; INPUT                                                         :
3686                                    <1> ;     (AL) = WRITE STRING COMMAND  0 - 3                                :
3687                                    <1> ;     (BH) = DISPLAY PAGE (ACTIVE PAGE)                                 :
3688                                    <1> ;     (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN              :
3689                                    <1> ;     (DX) = CURSOR POSITION FOR START OF STRING WRITE                   :
3690                                    <1> ;     (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1       :
3691                                    <1> ;     (eBP) = SOURCE STRING OFFSET                                      :
3692                                    <1> ; OUTPUT                                                        :
3693                                    <1> ;     NONE                                                      :
3694                                    <1> ;-----------------------------------------------------------------------------
3695                                    <1>
3696                                    <1> ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
3697                                    <1> ; AL = 01h: Assign all characters the attribute in BL; update cursor
3698                                    <1> ; AL = 02h: Use attributes in string; do not update cursor
3699                                    <1> ; AL = 03h: Use attributes in string; update cursor
3700                                    <1>
3701                                    <1> WRITE_STRING:
3702                                    <1>       ; 12/09/2016
3703                                    <1>       ; 09/07/2016
3704                                    <1>       ;cmp   byte [CRT_MODE], 7 ; 6?!
3705                                    <1>       ;ja    VIDEO_RETURN        ; not a valid function for VGA modes
3706                                    <1>       ;
3707 00002463 A2[AC960100]             <1>       mov    [w_str_cmd], al           ; save (AL) command
3708 00002468 3C04                     <1>       CMP    AL, 4               ; TEST FOR INVALID WRITE STRING OPTION
3709 0000246A 0F83EFF6FFFF             <1>       JNB    VIDEO_RETURN        ; IF OPTION INVALID THEN RETURN
3710                                    <1>
3711                                    <1>         ;JCXZ   VIDEO_RETURN            ; IF ZERO LENGTH STRING THEN RETURN
3712                                    <1>
3713 00002470 67E362                   <1>         jcxz    P55                   ; 01/07/2016
3714                                    <1>
3715                                    <1>       ; 01/07/2016
3716                                    <1>       ;and   ecx, 0FFFFh
3717                                    <1>       ; ECX = byte count
3718                                    <1>       ;push  ecx
3719 00002473 89EE                     <1>       mov    esi, ebp ; user buffer
3720 00002475 BF00000700               <1>       mov    edi, Cluster_Buffer  ; system buffer
3721 0000247A E812F70000               <1>       call   transfer_from_user_buffer
3722                                    <1>       ;pop   ecx
3723 0000247F 0F82DAF6FFFF             <1>       jc     VIDEO_RETURN
3724                                    <1>       ; ecx = transfer (byte) count = character count
3725 00002485 BD00000700               <1>       mov    ebp, Cluster_Buffer
3726                                    <1>       ; 12/09/2016
3727 0000248A 803D[DA6F0000]07         <1>       cmp    byte [CRT_MODE], 7 ; 6?!
3728 00002491 0F87A1000000             <1>       ja     vga_write_string
```

```
                                <1>       ;
00002497 0FB6F7                 <1>       movzx  esi, bh                  ; GET CURRENT CURSOR PAGE
0000249A 66D1E6                 <1>       SAL    SI, 1                ; CONVERT TO PAGE OFFSET  (SI= PAGE)
                                <1>       ; *****
0000249D 66FFB6[368A0100]       <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:
000024A4 51                     <1>       push   ecx ; ****
000024A5 53                     <1>       push   ebx ; *** ; 18/11/2020
000024A6 56                     <1>       push   esi ; **
000024A7 52                     <1>       push   edx ; *
000024A8 E8F2FEFFFF             <1>       call   _set_cpos
                                <1> P50:
000024AD 8A4500                 <1>       MOV    AL, [eBP]          ; GET CHARACTER FROM INPUT STRING
000024B0 45                     <1>       INC    eBP                ; BUMP POINTER TO CHARACTER
                                <1>
                                <1> ;-----       TEST FOR SPECIAL CHARACTER'S
                                <1>
000024B1 3C08                   <1>       CMP    AL, 08H                  ; IS IT A BACKSPACE
000024B3 7410                   <1>       JE     short P51          ; BACK_SPACE
000024B5 3C0D                   <1>       CMP    AL, 0Dh ; CR       ; IS IT CARRIAGE RETURN
000024B7 740C                   <1>       JE     short P51          ; CAR_RET
000024B9 3C0A                   <1>       CMP    AL, 0Ah ; LF       ; IS IT A LINE FEED
000024BB 7408                   <1>       JE     short P51          ; LINE_FEED
                                <1>       ; 18/11/2020
000024BD 3C09                   <1>       cmp    al, 09h                  ; is it a tab stop
000024BF 7404                   <1>       je     short P51
                                <1>       ;
000024C1 3C07                   <1>       CMP    AL, 07h                  ; IS IT A BELL
000024C3 7515                   <1>       JNE    short P52          ; IF NOT THEN DO WRITE CHARACTER
                                <1> P51:
                                <1>       ;MOV   AH,0EH             ; TTY_CHARACTER_WRITE
                                <1>       ;INT   10H                ; WRITE TTY CHARACTER TO THE CRT
                                <1>
000024C5 E84EFEFFFF             <1>       call   _write_tty_m3
                                <1>
000024CA 5A                     <1>       pop    edx ; *
000024CB 5E                     <1>       pop    esi ; **
                                <1>
000024CC 668B96[368A0100]       <1>       MOV    DX, [eSI+CURSOR_POSN]     ; GET CURRENT CURSOR POSITION
000024D3 EB44                   <1>       JMP    SHORT P54          ; SET CURSOR POSITION AND CONTINUE
                                <1> P55:
000024D5 E985F6FFFF             <1>       JMP    VIDEO_RETURN
                                <1> P52:
000024DA 66B90100               <1>       MOV    CX, 1              ; SET CHARACTER WRITE AMOUNT TO ONE
000024DE 803D[AC960100]02       <1>       CMP    byte [w_str_cmd], 2 ; IS THE ATTRIBUTE IN THE STRING
000024E5 7204                   <1>       JB     short P53          ; IF NOT THEN SKIP
000024E7 8A5D00                 <1>       MOV    BL, [eBP]          ; ELSE GET NEW ATTRIBUTE
000024EA 45                     <1>       INC    eBP                ; BUMP STRING POINTER
                                <1> P53:
                                <1>       ;MOV   AH,09H             ; GOT_CHARACTER
                                <1>       ;INT   10H                ; WRITE CHARACTER TO THE CRT
                                <1>
000024EB E88EFDFFFF             <1>       call   _write_c_current
                                <1>
000024F0 5A                     <1>       pop    edx ; *
                                <1>
                                <1>       ; 05/12/2020
                                <1>       ; bx is preserved in '_write_c_current'
                                <1>       ; 18/11/2020
                                <1>       ;mov   ebx, [esp+4] ; ***
                                <1>
000024F1 0FB6F7                 <1>       movzx  esi, bh ; video page number (0 to 7)
000024F4 889E[E36F0000]         <1>       mov    [esi+chr_attrib], bl ; color/attribute
                                <1>
000024FA FEC2                   <1>       INC    DL                 ; INCREMENT COLUMN COUNTER
000024FC 3A15[DC6F0000]         <1>       CMP    DL, [CRT_COLS]           ; IF COLS ARE WITHIN RANGE FOR THIS MODE
                                <1>       ;JB    short P54          ;    THEN GO TO COLUMNS SET
00002502 7214                   <1>       jb     short P56 ; 05/12/2020
00002504 FEC6                   <1>       INC    DH                 ; BUMP ROW COUNTER BY ONE
00002506 28D2                   <1>       SUB    DL, DL             ; SET COLUMN COUNTER TO ZERO
00002508 80FE19                 <1>       CMP    DH, 25             ; IF ROWS ARE LESS THAN 25 THEN
                                <1>       ;JB    short P54          ; GO TO ROWS_COLUMNS_SET
0000250B 720B                   <1>       jb     short P56 ; 05/12/2020
                                <1>
                                <1>       ; 18/11/2020
                                <1>       ;MOV   AX,0E0AH           ; ELSE SCROLL SCREEN
                                <1>       ;INT   10H                ; RESET ROW COUNTER TO 24
                                <1>
                                <1>       ; 18/11/2020
0000250D B00A                   <1>       mov    al, 0Ah      ; line feed
                                <1>
0000250F E804FEFFFF             <1>       call   _write_tty_m3
                                <1>
00002514 66BA0018               <1>       mov    dx, 1800h          ; Column = 0, Row = 24
                                <1> P56:
                                <1>       ; 05/12/2020
                                <1>       ; 18/11/2020
00002518 5E                     <1>       pop    esi ; **
                                <1> P54:                             ; ROW_COLUMNS_SET
                                <1>       ;MOV   AX,0200H           ; SET NEW CURSOR POSITION COMMAND
                                <1>       ;INT   10H                ; ESTABLISH NEW CURSOR POSITION
                                <1>
                                <1>       ; 18/11/2020
00002519 5B                     <1>       pop    ebx ; ***
0000251A 59                     <1>       pop    ecx ; ****
                                <1>
                                <1>       ;LOOP  P50                ; DO IT ONCE MORE UNTIL (CX) = ZERO
0000251B 6649                   <1>       dec    cx
0000251D 7585                   <1>       jnz    short P50next
                                <1>
0000251F 665A                   <1>       POP    DX  ; *****        ; RESTORE OLD CURSOR COORDINATES
                                <1>
```

```
3834 00002521 F605[AC960100]01    <1>       test   byte [w_str_cmd], 1 ; IF CURSOR WAS NOT TO BE MOVED
3835 00002528 0F8531F6FFFF        <1>       JNZ    VIDEO_RETURN        ; THEN EXIT WITHOUT RESETTING OLD VALUE
3836                              <1>
3837                              <1>       ;MOV   AX,0200H            ; ELSE RESTORE OLD CURSOR POSITION
3838                              <1>       ;INT   10H
3839                              <1>                                   ; DONE - EXIT WRITE STRING
3840 0000252E E86CFEFFFF          <1>       call   _set_cpos
3841 00002533 E927F6FFFF          <1>       JMP    VIDEO_RETURN        ; RETURN TO CALLER
3842                              <1>
3843                              <1> vga_write_string:
3844                              <1>       ; 12/09/2016 - TRDOS 386 (TRDOS v2.0)
3845                              <1>       ;
3846                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
3847                              <1>       ; vgabios-0.7a (2011)
3848                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
3849                              <1>       ; 'vgabios.c', ' biosfn_write_string'
3850                              <1>
3851                              <1>       ; INPUT                                                 :
3852                              <1>       ;      (AL) = WRITE STRING COMMAND  0 - 3                       :
3853                              <1>       ;      (BH) = DISPLAY PAGE (ACTIVE PAGE)                        :
3854                              <1>       ;      (CX) = COUNT OF CHARACTERS TO WRITE, IF (CX) = 0 THEN RETURN      :
3855                              <1>       ;      (DX) = CURSOR POSITION FOR START OF STRING WRITE          :
3856                              <1>       ;      (BL) = ATTRIBUTE OF CHARACTER TO WRITE IF (AL) = 0  OR     (AL) = 1  :
3857                              <1>       ;      (eBP) = SOURCE STRING OFFSET                             :
3858                              <1>       ; OUTPUT                                                :
3859                              <1>       ;      NONE                                             :
3860                              <1>       ;-------------------------------------------------------------------------;
3861                              <1>
3862                              <1>       ; AL = 00h: Assign all characters the attribute in BL; do not update cursor
3863                              <1>       ; AL = 01h: Assign all characters the attribute in BL; update cursor
3864                              <1>       ; AL = 02h: Use attributes in string; do not update cursor
3865                              <1>       ; AL = 03h: Use attributes in string; update cursor
3866                              <1>
3867                              <1>       ; biosfn_write_string(GET_AL(),GET_BH(),GET_BL(),CX,GET_DH(),GET_DL(),ES,BP);
3868                              <1>       ; static void biosfn_write_string (flag,page,attr,count,row,col,seg,offset)
3869                              <1>
3870                              <1>       ; // Read curs info for the page
3871                              <1>       ; biosfn_get_cursor_pos(page,&dummy,&oldcurs);
3872                              <1>       ; bh = video page = 0
3873                              <1>       ;movzx esi, word [CURSOR_POSN] ; current cursor position for video page 0
3874                              <1>
3875                              <1>       ; // if row=0xff special case : use current cursor position
3876                              <1>       ; if(row==0xff)
3877                              <1>       ;  {col=oldcurs&0x00ff;
3878                              <1>       ;   row=(oldcurs&0xff00)>>8;
3879                              <1>       ;  }
3880                              <1>
3881                              <1>       ;mov   al, [w_str_cmd]
3882                              <1>
3883 00002538 80FEFF              <1>       cmp    dh, 0FFh
3884 0000253B 7407                <1>       je     short vga_wstr_1 ; user current cursor position
3885                              <1> vga_wstr_0:
3886                              <1>       ; set cursor position
3887 0000253D 668915[368A0100]    <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
3888                              <1> vga_wstr_1:
3889 00002544 66FF35[368A0100]    <1>       push   word [CURSOR_POSN] ; *
3890                              <1>
3891                              <1>       ; ebp = string offset in system buffer (user buffer was copied to)
3892                              <1>
3893                              <1>       ; while(count--!=0)
3894                              <1>       ;  {
3895                              <1>       ;   car=read_byte(seg,offset++);
3896                              <1>       ;   if((flag&0x02)!=0)
3897                              <1>       ;    attr=read_byte(seg,offset++);
3898                              <1>       ;    biosfn_write_teletype(car,page,attr,WITH_ATTR);
3899                              <1>       ;  }
3900                              <1>
3901                              <1>       ;push  eax ; **
3902                              <1>       ;test  al, 2
3903 0000254B F605[AC960100]02    <1>       test   byte [w_str_cmd], 2
3904 00002552 751D                <1>       jnz    short vga_wstr_3
3905 00002554 881D[478A0100]      <1>       mov    [ccolor], bl
3906                              <1> vga_wstr_2:
3907 0000255A 51                  <1>       push   ecx
3908 0000255B 8A4500              <1>       mov    al, [ebp]
3909 0000255E E8D90A0000          <1>       call   vga_write_teletype
3910 00002563 59                  <1>       pop    ecx
3911 00002564 6649                <1>       dec    cx
3912 00002566 741E                <1>       jz     short vga_wstr_4
3913 00002568 45                  <1>       inc    ebp
3914 00002569 8A1D[478A0100]      <1>       mov    bl, [ccolor]
3915 0000256F EBE9                <1>       jmp    short vga_wstr_2
3916                              <1> vga_wstr_3:
3917 00002571 51                  <1>       push   ecx
3918 00002572 8A4500              <1>       mov    al, [ebp]
3919 00002575 45                  <1>       inc    ebp
3920 00002576 8A5D00              <1>       mov    bl, [ebp]
3921 00002579 E8BE0A0000          <1>       call   vga_write_teletype
3922 0000257E 59                  <1>       pop    ecx
3923 0000257F 6649                <1>       dec    cx
3924 00002581 7403                <1>       jz     short vga_wstr_4
3925 00002583 45                  <1>       inc    ebp
3926 00002584 EBEB                <1>       jmp    short vga_wstr_3
3927                              <1> vga_wstr_4:
3928                              <1>       ; // Set back curs pos
3929                              <1>       ; if((flag&0x01)==0)
3930                              <1>       ;  biosfn_set_cursor_pos(page,oldcurs);
3931                              <1>       ; }
3932                              <1>       ;pop   eax ; **
3933 00002586 665A                <1>       pop    dx ; word [CURSOR_POSN] ; *
3934                              <1>       ;test  al, 1
3935 00002588 F605[AC960100]01    <1>       test   byte [w_str_cmd], 1
3936 0000258F 0F85CAF5FFFF        <1>       jnz    VIDEO_RETURN
3937 00002595 668915[368A0100]    <1>       mov    [CURSOR_POSN], dx
3938 0000259C E9BEF5FFFF          <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]
000025A1 80FC07                 <1>       cmp    ah, 7
000025A4 7766                   <1>       ja     short vga_graphics_up
                                <1>       ;je    n0
                                <1>
000025A6 88C7                   <1>       MOV    bh, al              ; save line count in BH
000025A8 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>
000025AB E8DA050000             <1>       CALL   GRAPH_POSN
000025B0 0FB7F8                 <1>       MOVzx  eDI, AX                    ; SAVE RESULT AS DESTINATION ADDRESS
                                <1>
                                <1> ;-----      DETERMINE SIZE OF WINDOW
                                <1>
000025B3 6629CA                 <1>       SUB    DX, CX
000025B6 6681C20101             <1>         ADD     DX, 101h                ; ADJUST VALUES
000025BB C0E602                 <1>       SAL    DH, 2               ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
                                <1>                                  ; AND EVEN/ODD ROWS
                                <1> ;-----      DETERMINE CRT MODE
                                <1>
000025BE 803D[DA6F0000]06       <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
000025C5 7305                   <1>         JNC     short _R7_              ; FIND_SOURCE
                                <1>
                                <1> ;-----      MEDIUM RES UP
000025C7 D0E2                   <1>       SAL    DL, 1               ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
000025C9 66D1E7                 <1>       SAL    DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
                                <1>
                                <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
                                <1> _R7_:                                   ; FIND_SOURCE
000025CC 81C700800B00           <1>       add    edi, 0B8000h
000025D2 C0E702                 <1>       sal    bh, 2               ; multiply number of lines by 4
000025D5 7431                   <1>         JZ      short _R11              ; IF ZERO, THEN BLANK ENTIRE FIELD
000025D7 B050                   <1>       MOV    AL, 80              ; 80 BYTES/ROW
000025D9 F6E7                   <1>       mul    bh                  ; determine offset to source
000025DB 0FB7F0                 <1>       movzx  esi, ax                   ; offset to source
000025DE 01FE                   <1>       add    eSI, eDI            ; SET UP SOURCE
000025E0 88F4                   <1>       MOV    AH, DH              ; NUMBER OF ROWS IN FIELD
000025E2 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
000025E4 E813040000             <1>         CALL    _R17                    ; MOVE ONE ROW
000025E9 6681EEB01F             <1>       SUB    SI, 2000h-80         ; MOVE TO NEXT ROW
000025EE 6681EFB01F             <1>       SUB    DI, 2000h-80
000025F3 FECC                   <1>       DEC    AH                   ; NUMBER OF ROWS TO MOVE
000025F5 75ED                   <1>         JNZ     short _R8               ; CONTINUE TILL ALL MOVED
                                <1>
                                <1> ;-----      FILL IN THE VACATED LINE(S)
                                <1> _R9:                                    ; CLEAR ENTRY
000025F7 88D8                   <1>       mov    al, bl              ; attribute to fill with
                                <1> _R10_:
000025F9 E81A040000             <1>         CALL    _R18                    ; CLEAR THAT ROW
000025FE 6681EFB01F             <1>       SUB    DI, 2000h-80         ; POINT TO NEXT LINE
00002603 FECF                   <1>       dec    bh                   ; number of lines to fill
00002605 75F2                   <1>         JNZ     short _R10_             ; CLEAR LOOP
00002607 C3                     <1>       retn                        ; EVERYYHING DONE
                                <1>
                                <1> _R11:                                   ; BLANK_FIELD
00002608 88F7                   <1>       mov    bh, dh              ; set blank count to everything in field
0000260A EBEB                   <1>         JMP     short _R9               ; CLEAR THE FIELD
                                <1>
                                <1> vga_graphics_up:
                                <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'
```

```
4044                                    <1>       ;
4045                                    <1>
4046                                    <1>       ; cl = upper left column
4047                                    <1>       ; ch = upper left row
4048                                    <1>       ; dl = lower rigth column
4049                                    <1>       ; dh = lower right row
4050                                    <1>       ;
4051                                    <1>       ; al = line count (AL=0 means blank entire fields)
4052                                    <1>       ; bl = fill value for blanked lines
4053                                    <1>       ; bh = unused
4054                                    <1>       ;
4055                                    <1>       ; ah = [CRT_MODE], current video mode
4056                                    <1>
4057 0000260C 88C7                     <1>       mov   bh, al ; 31/07/2016
4058 0000260E BE[FE6F0000]             <1>       mov   esi, vga_g_modes
4059 00002613 89F7                     <1>       mov   edi, esi
4060 00002615 83C708                   <1>       add   edi, vga_g_mode_count
4061                                    <1> vga_g_up_0:
4062 00002618 AC                       <1>       lodsb
4063 00002619 38E0                     <1>       cmp   al, ah ; [CRT_MODE]
4064 0000261B 7405                     <1>       je    short vga_g_up_1
4065 0000261D 39FE                     <1>       cmp   esi, edi
4066 0000261F 72F7                     <1>       jb    short vga_g_up_0
4067                                    <1>       ;xor  bh, bh ; 31/07/2016)
4068 00002621 C3                       <1>       retn  ; nothing to do
4069                                    <1> vga_g_up_1:
4070 00002622 88F8                     <1>       mov   al, bh ; 31/07/2016
4071 00002624 83C64F                   <1>       add   esi, vga_g_memmodel - (vga_g_modes + 1)
4072                                    <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
4073                                    <1>
4074                                    <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
4075                                    <1>       ; if(clr>=nbcols)clr=nbcols-1;
4076                                    <1>       ; if(nblines>nbrows)nblines=0;
4077                                    <1>       ; cols=clr-cul+1;
4078                                    <1>
4079 00002627 3A35[E26F0000]           <1>       cmp   dh, [VGA_ROWS]
4080 0000262D 7208                     <1>       jb    short vga_g_up_2
4081 0000262F 8A35[E26F0000]           <1>       mov   dh, [VGA_ROWS]
4082 00002635 FECE                     <1>       dec   dh
4083                                    <1> vga_g_up_2:
4084 00002637 3A15[DC6F0000]           <1>       cmp   dl, [CRT_COLS]  ; = [VGA_COLS]
4085 0000263D 7208                     <1>       jb    short vga_g_up_3
4086 0000263F 8A15[DC6F0000]           <1>       mov   dl, [CRT_COLS]
4087 00002645 FECA                     <1>       dec   dl
4088                                    <1> vga_g_up_3:
4089 00002647 3A05[E26F0000]           <1>       cmp   al, [VGA_ROWS]
4090 0000264D 7602                     <1>       jna   short vga_g_up_4
4091 0000264F 28C0                     <1>       sub   al, al ; 0
4092                                    <1> vga_g_up_4:
4093 00002651 88D7                     <1>       mov   bh, dl ; clr
4094 00002653 28CF                     <1>       sub   bh, cl ; cul
4095 00002655 FEC7                     <1>       inc   bh ; cols = clr-cul+1
4096                                    <1>
4097 00002657 20C0                     <1>       and   al, al ; nblines = 0
4098 00002659 755D                     <1>       jnz   short vga_g_up_6
4099 0000265B 20ED                     <1>       and   ch, ch ; rul = 0
4100 0000265D 7559                     <1>       jnz   short vga_g_up_6
4101 0000265F 20C9                     <1>       and   cl, cl ; cul = 0
4102 00002661 7555                     <1>       jnz   short vga_g_up_6
4103                                    <1>
4104 00002663 6650                     <1>       push  ax
4105 00002665 A0[E26F0000]             <1>       mov   al, [VGA_ROWS]
4106 0000266A FEC8                     <1>       dec   al
4107 0000266C 38C6                     <1>       cmp   dh, al ; rlr = nbrows-1
4108 0000266E 7546                     <1>       jne   short vga_g_up_5
4109 00002670 A0[DC6F0000]             <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
4110 00002675 FEC8                     <1>       dec   al
4111 00002677 38C2                     <1>       cmp   dl, al ; clr = nbcols-1
4112 00002679 753B                     <1>       jne   short vga_g_up_5
4113 0000267B 6658                     <1>       pop   ax
4114                                    <1>
4115 0000267D 66B80502                 <1>       mov   ax, 0205h
4116 00002681 66BACE03                 <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4117 00002685 66EF                     <1>       out   dx, ax
4118 00002687 A0[E26F0000]             <1>       mov   al, [VGA_ROWS]
4119 0000268C 8A25[DC6F0000]           <1>       mov   ah, [CRT_COLS] ; = [VGA_COLS]
4120 00002692 F6E4                     <1>       mul   ah
4121 00002694 0FB7D0                   <1>       movzx edx, ax
4122                                    <1>       ; 08/08/2016
4123 00002697 0FB605[DE6F0000]         <1>       movzx eax, byte [CHAR_HEIGHT]
4124 0000269E F7E2                     <1>       mul   edx
4125                                    <1>       ; eax = byte count
4126 000026A0 89C1                     <1>       mov   ecx, eax
4127                                    <1>       ;; 07/08/2016
4128                                    <1>       ;shl  dx, 3 ; * 8 ; * [CHAR_HEIGHT]
4129                                    <1>       ;mov  ecx, edx
4130 000026A2 88D8                     <1>       mov   al, bl ; fill value for blanked lines
4131 000026A4 BF00000A00               <1>       mov   edi, 0A0000h
4132 000026A9 F3AA                     <1>       rep   stosb
4133                                    <1>
4134 000026AB 66B80500                 <1>       mov   ax, 5
4135 000026AF 66BACE03                 <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4136 000026B3 66EF                     <1>       out   dx, ax ; 0005h
4137                                    <1>
4138 000026B5 C3                       <1>       retn
4139                                    <1>
4140                                    <1> vga_g_up_5:
4141 000026B6 6658                     <1>       pop   ax
4142                                    <1>
4143                                    <1> vga_g_up_6:
4144                                    <1>       ; [ESI] = VGA memory model number for current video mode
4145                                    <1>         ;
4146                                    <1>         ; LINEAR8 equ 5
4147                                    <1>         ; PLANAR4 equ 4
4148                                    <1>         ; PLANAR1 equ 3
```

```
4149                                    <1>
4150 000026B8 803E04                    <1>       cmp    byte [esi], PLANAR4
4151 000026BB 7424                      <1>       je     short vga_g_up_planar
4152 000026BD 803E03                    <1>       cmp    byte [esi], PLANAR1
4153 000026C0 741F                      <1>       je     short vga_g_up_planar
4154                                    <1> vga_g_up_linear8:
4155                                    <1>       ; 07/07/2016 (TEMPORARY)
4156                                    <1>       ;
4157                                    <1>       ; cl = upper left column ; cul
4158                                    <1>       ; ch = upper left row ; rul
4159                                    <1>       ; dl = lower rigth column ; clr
4160                                    <1>       ; dh = lower right row ; rlr
4161                                    <1>
4162                                    <1> vga_g_up_l0:
4163                                    <1>       ;{for(i=rul;i<=rlr;i++)
4164                                    <1>       ; if((i+nblines>rlr)||(nblines==0))
4165 000026C2 08C0                      <1>       or     al, al
4166 000026C4 7414                      <1>       jz     short vga_g_up_l2
4167 000026C6 88C4                      <1>       mov    ah, al
4168 000026C8 00EC                      <1>       add    ah, ch ; i+nblines
4169                                    <1>       ;jc    short vga_g_up_l2
4170 000026CA 38F4                      <1>       cmp    ah, dh
4171 000026CC 770C                      <1>       ja     short vga_g_up_l2
4172                                    <1>       ; else
4173                                    <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
4174 000026CE E8F2000000                <1>       call   vgamem_copy_l8
4175                                    <1> vga_g_up_l1:
4176 000026D3 FEC5                      <1>       inc    ch
4177 000026D5 38F5                      <1>       cmp    ch, dh
4178 000026D7 76E9                      <1>       jna    short vga_g_up_l0
4179 000026D9 C3                        <1>       retn
4180                                    <1> vga_g_up_l2:
4181                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4182 000026DA E850010000                <1>       call   vgamem_fill_l8
4183 000026DF EBF2                      <1>       jmp    short vga_g_up_l1
4184                                    <1>
4185                                    <1> vga_g_up_planar:
4186                                    <1>       ; cl = upper left column ; cul
4187                                    <1>       ; ch = upper left row ; rul
4188                                    <1>       ; dl = lower rigth column ; clr
4189                                    <1>       ; dh = lower right row ; rlr
4190                                    <1> vga_g_up_pl0:
4191                                    <1>       ;{for(i=rul;i<=rlr;i++)
4192                                    <1>       ; if((i+nblines>rlr)||(nblines==0))
4193 000026E1 20C0                      <1>       and    al, al
4194 000026E3 7414                      <1>       jz     short vga_g_up_pl2
4195 000026E5 88C4                      <1>       mov    ah, al
4196 000026E7 00EC                      <1>       add    ah, ch ; i+nblines
4197                                    <1>       ;jc    short vga_g_up_pl2
4198 000026E9 38F4                      <1>       cmp    ah, dh
4199 000026EB 770C                      <1>       ja     short vga_g_up_pl2
4200                                    <1>       ; else
4201                                    <1>       ;  vgamem_copy_pl4(cul,i+nblines,i,cols,nbcols,cheight);
4202 000026ED E80E000000                <1>       call   vgamem_copy_pl4
4203                                    <1> vga_g_up_pl1:
4204 000026F2 FEC5                      <1>       inc    ch
4205 000026F4 38F5                      <1>       cmp    ch, dh
4206 000026F6 76E9                      <1>       jna    short vga_g_up_pl0
4207 000026F8 C3                        <1>       retn
4208                                    <1> vga_g_up_pl2:
4209                                    <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
4210 000026F9 E870000000                <1>       call   vgamem_fill_pl4
4211 000026FE EBF2                      <1>       jmp    short vga_g_up_pl1
4212                                    <1>
4213                                    <1> vgamem_copy_pl4:
4214                                    <1>       ; 08/08/2016
4215                                    <1>       ; 07/08/2016
4216                                    <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4217                                    <1>       ;
4218                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
4219                                    <1>       ; vgabios-0.7a (2011)
4220                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
4221                                    <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
4222                                    <1>       ;
4223                                    <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
4224                                    <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
4225                                    <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
4226                                    <1>
4227                                    <1>       ; src=ysrc*cheight*nbcols+xstart;
4228                                    <1>       ; dest=ydest*cheight*nbcols+xstart;
4229                                    <1>
4230 00002700 52                      <1>       push   edx
4231 00002701 50                      <1>       push   eax
4232                                    <1>
4233                                    <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
4234 00002702 66B80501                  <1>       mov    ax, 0105h
4235 00002706 66BACE03                  <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4236 0000270A 66EF                      <1>       out    dx, ax
4237                                    <1>
4238                                    <1>       ; 07/08/2016
4239                                    <1>       ;mov     ah, [esp+1]
4240                                    <1>       ;movzx edx, ah ; ysrc
4241 0000270C 0FB6542401                <1>       movzx  edx, byte [esp+1]
4242                                    <1>       ; 08/08/2016
4243 00002711 0FB605[DE6F0000]          <1>       movzx  eax, byte [CHAR_HEIGHT]
4244 00002718 8A25[DC6F0000]            <1>       mov    ah, [CRT_COLS] ; nbcols
4245 0000271E F6E4                      <1>       mul    ah
4246                                    <1>       ;; 07/08/2016
4247                                    <1>       ;movzx eax, byte [CRT_COLS]
4248                                    <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4249 00002720 50                        <1>       push   eax ; cheight * nbcols
4250 00002721 F7E2                      <1>       mul    edx ; * ysrc
4251                                    <1>       ; eax = ysrc * cheight * nbcols
4252                                    <1>       ; edx = 0
4253 00002723 88CA                      <1>       mov    dl, cl ; edx = xstart
```

```
4254 00002725 01D0              <1>       add    eax, edx
4255 00002727 89C6              <1>       mov    esi, eax ; src
4256 00002729 88EA              <1>       mov    dl, ch ; ydest
4257 0000272B 58                <1>       pop    eax ; cheight * nbcols
4258 0000272C F7E2              <1>       mul    edx
4259                            <1>       ; eax = ydest * cheight * nbcols
4260 0000272E 88CA              <1>       mov    dl, cl ; edx = xstart
4261 00002730 01D0              <1>       add    eax, edx
4262 00002732 89C7              <1>       mov    edi, eax ; dest
4263                            <1>       ; esi = src
4264                            <1>       ; edi = dest
4265                            <1>       ; for(i=0;i<cheight;i++)
4266                            <1>       ; {
4267                            <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
4268                            <1>       ; }
4269 00002734 51                <1>       push   ecx
4270 00002735 B900000A00        <1>       mov    ecx, 0A0000h
4271 0000273A 01CE              <1>       add    esi, ecx
4272 0000273C 01CF              <1>       add    edi, ecx
4273                            <1>       ; 08/08/2016
4274 0000273E 8A35[DE6F0000]    <1>       mov    dh, [CHAR_HEIGHT]
4275                            <1>       ;; 07/08/2016
4276                            <1>       ;mov   dh, 8 ; 07/08/2016
4277 00002744 28D2              <1>       sub    dl, dl ; i
4278                            <1> vgamem_copy_pl4_0:
4279 00002746 56                <1>       push   esi
4280 00002747 57                <1>       push   edi
4281 00002748 0FB605[DC6F0000]  <1>       movzx  eax, byte [CRT_COLS]
4282 0000274F F6E2              <1>       mul    dl
4283                            <1>       ; eax = i * nbcols
4284 00002751 01C7              <1>       add    edi, eax ; dest+i*nbcols
4285 00002753 01C6              <1>       add    esi, eax
4286 00002755 0FB6CF            <1>       movzx  ecx, bh ; cols
4287 00002758 F3A4              <1>       rep    movsb
4288 0000275A 5F                <1>       pop    edi
4289 0000275B 5E                <1>       pop    esi
4290 0000275C FECE              <1>       dec    dh
4291 0000275E 75E6              <1>       jnz    short vgamem_copy_pl4_0
4292                            <1> vgamem_copy_pl4_1:
4293 00002760 59                <1>       pop    ecx
4294                            <1>
4295                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4296 00002761 66B80500          <1>       mov    ax, 0005h
4297 00002765 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4298 00002769 66EF              <1>       out    dx, ax
4299                            <1>
4300 0000276B 58                <1>       pop    eax
4301 0000276C 5A                <1>       pop    edx
4302                            <1>
4303 0000276D C3                <1>       retn
4304                            <1>
4305                            <1> vgamem_fill_pl4:
4306                            <1>       ; 08/08/2016
4307                            <1>       ; 07/08/2016
4308                            <1>       ; 04/08/2016
4309                            <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4310                            <1>       ;
4311                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
4312                            <1>       ; vgabios-0.7a (2011)
4313                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
4314                            <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
4315                            <1>       ;
4316                            <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
4317                            <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
4318                            <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
4319                            <1>
4320                            <1>       ; dest=ystart*cheight*nbcols+xstart;
4321 0000276E 52                <1>       push   edx
4322 0000276F 50                <1>       push   eax
4323                            <1>
4324                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205)
4325 00002770 66B80502          <1>       mov    ax, 0205h
4326 00002774 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
4327 00002778 66EF              <1>       out    dx, ax
4328                            <1>
4329                            <1>             ; 08/08/2016
4330 0000277A 0FB605[DE6F0000]  <1>       movzx   eax, byte [CHAR_HEIGHT]
4331 00002781 F6E5              <1>       mul    ch
4332                            <1>       ;; 07/08/2016
4333                            <1>       ;movzx eax, ch
4334                            <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4335 00002783 0FB615[DC6F0000]  <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
4336 0000278A F7E2              <1>       mul    edx
4337                            <1>       ; edx  = 0
4338 0000278C 88CA              <1>       mov    dl, cl
4339 0000278E 01D0              <1>       add    eax, edx
4340 00002790 89C7              <1>       mov    edi, eax
4341                            <1>       ; edi = dest
4342                            <1>       ; for(i=0;i<cheight;i++)
4343                            <1>       ; {
4344                            <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
4345                            <1>       ; }
4346 00002792 81C700000A00      <1>       add    edi, 0A0000h
4347 00002798 51                <1>       push   ecx
4348                            <1>       ; 08/08/2016
4349 00002799 8A35[DE6F0000]    <1>       mov    dh, [CHAR_HEIGHT]
4350                            <1>       ;; 07/08/2016
4351                            <1>       ;mov   dh, 8 ; 07/08/2016
4352 0000279F 28D2              <1>       sub    dl, dl ; i
4353                            <1> vgamem_fill_pl4_0:
4354 000027A1 57                <1>       push   edi
4355 000027A2 0FB605[DC6F0000]  <1>       movzx  eax, byte [CRT_COLS]
4356 000027A9 F6E2              <1>       mul    dl
4357                            <1>       ; eax = i * nbcols
4358 000027AB 01C7              <1>       add    edi, eax ; dest+i*nbcols
```

```
4359 000027AD 88D8                <1>       mov   al, bl ; attr ; 04/08/2016
4360 000027AF 0FB6CF              <1>       movzx ecx, bh ; cols
4361 000027B2 F3AA                <1>       rep   stosb
4362 000027B4 5F                  <1>       pop   edi
4363 000027B5 75EA                <1>       jnz   short vgamem_fill_pl4_0
4364                              <1> vgamem_fill_pl4_1:
4365 000027B7 59                  <1>       pop   ecx
4366                              <1>
4367                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4368 000027B8 66B80500            <1>       mov   ax, 0005h
4369 000027BC 66BACE03            <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4370 000027C0 66EF                <1>       out   dx, ax
4371                              <1>
4372 000027C2 58                  <1>       pop   eax
4373 000027C3 5A                  <1>       pop   edx
4374                              <1>
4375 000027C4 C3                  <1>       retn
4376                              <1>
4377                              <1> vgamem_copy_l8:
4378                              <1>       ; 08/08/2016
4379                              <1>       ; 07/08/2016
4380                              <1>       ; 06/08/2016
4381                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4382                              <1>       ;
4383                              <1>       ; TEMPORARY
4384                              <1>       ;
4385                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
4386                              <1>       ; vgabios-0.7a (2011)
4387                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
4388                              <1>       ; 'vgabios.c', 'vgamem_copy_pl4'
4389                              <1>       ;
4390                              <1>       ; vgamem_copy_pl4(xstart,ysrc,ydest,cols,nbcols,cheight)
4391                              <1>       ; cl = xstart, ah = ysrc (i+nblines), ch = ydest (i),
4392                              <1>       ; bh = cols, [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
4393                              <1>
4394                              <1>       ; src=ysrc*cheight*nbcols+xstart;
4395                              <1>       ; dest=ydest*cheight*nbcols+xstart;
4396                              <1>
4397 000027C5 52                  <1>       push  edx
4398 000027C6 50                  <1>       push  eax
4399                              <1>
4400                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0105)
4401                              <1>       ;mov  ax, 0105h
4402                              <1>       ;mov  dx, 3CEh ; VGAREG_GRDC_ADDRESS
4403                              <1>       ;out  dx, ax
4404                              <1>
4405                              <1>       ;mov  ah, [esp+1]
4406                              <1>
4407 000027C7 0FB6D4              <1>       movzx edx, ah ; ysrc
4408                              <1>       ; 08/08/2016
4409 000027CA 0FB605[DE6F0000]    <1>       movzx eax, byte [CHAR_HEIGHT]
4410 000027D1 8A25[DC6F0000]      <1>       mov   ah, [CRT_COLS] ; nbcols
4411 000027D7 F6E4                <1>       mul   ah
4412                              <1>       ;; 07/08/2016
4413                              <1>       ;movzx eax, byte [CRT_COLS]
4414                              <1>       ;shl  ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4415 000027D9 50                  <1>       push  eax ; cheight * nbcols
4416 000027DA F7E2                <1>       mul   edx ; * ysrc
4417                              <1>       ; eax = ysrc * cheight * nbcols
4418                              <1>       ; edx = 0
4419 000027DC 88CA                <1>       mov   dl, cl ; edx = xstart
4420 000027DE 01D0                <1>       add   eax, edx
4421 000027E0 89C6                <1>       mov   esi, eax ; src
4422 000027E2 66C1E603            <1>       shl   si, 3 ; * 8 ; 06/08/2016
4423 000027E6 88EA                <1>       mov   dl, ch ; ydest
4424 000027E8 58                  <1>       pop   eax ; cheight * nbcols
4425 000027E9 F7E2                <1>       mul   edx
4426                              <1>       ; eax = ydest * cheight * nbcols
4427 000027EB 88CA                <1>       mov   dl, cl ; edx = xstart
4428 000027ED 01D0                <1>       add   eax, edx
4429 000027EF 89C7                <1>       mov   edi, eax ; dest
4430 000027F1 66C1E703            <1>       shl   di, 3 ; * 8 ; 06/08/2016
4431                              <1>       ; esi = src
4432                              <1>       ; edi = dest
4433                              <1>       ; for(i=0;i<cheight;i++)
4434                              <1>       ; {
4435                              <1>       ;  memcpyb(0xa000,dest+i*nbcols,0xa000,src+i*nbcols,cols);
4436                              <1>       ; }
4437 000027F5 51                  <1>       push  ecx
4438 000027F6 B900000A00          <1>       mov   ecx, 0A0000h
4439 000027FB 01CE                <1>       add   esi, ecx
4440 000027FD 01CF                <1>       add   edi, ecx
4441                              <1>       ; 08/08/2016
4442 000027FF 8A35[DE6F0000]      <1>       mov   dh, [CHAR_HEIGHT]
4443                              <1>       ;; 07/08/2016
4444                              <1>       ;mov  dh, 8 ; 07/08/2016
4445 00002805 28D2                <1>       sub   dl, dl ; i
4446                              <1> vgamem_copy_l8_0:
4447 00002807 56                  <1>       push  esi
4448 00002808 57                  <1>       push  edi
4449 00002809 0FB605[DC6F0000]    <1>       movzx eax, byte [CRT_COLS]
4450 00002810 F6E2                <1>       mul   dl
4451                              <1>       ; eax = i * nbcols
4452 00002812 66C1E003            <1>       shl   ax, 3 ; * 8 ; 06/08/2016
4453 00002816 01C7                <1>       add   edi, eax ; dest+i*nbcols
4454 00002818 01C6                <1>       add   esi, eax
4455 0000281A 0FB6CF              <1>       movzx ecx, bh ; cols
4456 0000281D 66C1E103            <1>       shl   cx, 3 ; * 8 ; 06/08/2016
4457 00002821 F3A4                <1>       rep   movsb
4458 00002823 5F                  <1>       pop   edi
4459 00002824 5E                  <1>       pop   esi
4460 00002825 FEC2                <1>       inc   dl ; 06/08/2016
4461 00002827 FECE                <1>       dec   dh
4462 00002829 75DC                <1>       jnz   short vgamem_copy_l8_0
4463                              <1> vgamem_copy_l8_1:
```

```
4464 0000282B 59                  <1>       pop    ecx
4465                              <1>
4466                              <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4467                              <1>       ;mov   ax, 0005h
4468                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4469                              <1>       ;out   dx, ax
4470                              <1>
4471 0000282C 58                  <1>       pop    eax
4472 0000282D 5A                  <1>       pop    edx
4473                              <1>
4474 0000282E C3                  <1>       retn
4475                              <1>
4476                              <1> vgamem_fill_l8:
4477                              <1>       ; 08/08/2016
4478                              <1>       ; 07/08/2016
4479                              <1>       ; 06/08/2016
4480                              <1>       ; 04/08/2016
4481                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4482                              <1>       ;
4483                              <1>       ; TEMPORARY
4484                              <1>       ;
4485                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
4486                              <1>       ; vgabios-0.7a (2011)
4487                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
4488                              <1>       ; 'vgabios.c', 'vgamem_fill_pl4'
4489                              <1>       ;
4490                              <1>       ; vgamem_fill_pl4(xstart,ystart,cols,nbcols,cheight,attr)
4491                              <1>       ; cl = xstart, edi = ch = ystart, bh = cols,
4492                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight, attr = 0
4493                              <1>
4494                              <1>       ; dest=ystart*cheight*nbcols+xstart;
4495 0000282F 52                  <1>       push   edx
4496 00002830 50                  <1>       push   eax
4497                              <1>
4498                              <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0205)
4499                              <1>       ;mov   ax, 0205h
4500                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4501                              <1>       ;out   dx, ax
4502                              <1>
4503                              <1>         ; 08/08/2016
4504 00002831 0FB605[DE6F0000]    <1>       movzx   eax, byte [CHAR_HEIGHT]
4505 00002838 F6E5                <1>       mul    ch
4506                              <1>       ;; 07/08/2016
4507                              <1>       ;movzx eax, ch
4508                              <1>       ;shl   ax, 3 ; * 8 ; * [CHAR_HEIGHT]
4509 0000283A 0FB615[DC6F0000]    <1>       movzx  edx, byte [CRT_COLS] ; = [VGA_COLS]
4510 00002841 F7E2                <1>       mul    edx
4511                              <1>       ; edx  = 0
4512 00002843 88CA                <1>       mov    dl, cl
4513 00002845 01D0                <1>       add    eax, edx
4514 00002847 89C7                <1>       mov    edi, eax
4515 00002849 66C1E703            <1>       shl    di, 3 ; * 8 ; 06/08/2016
4516                              <1>       ; edi = dest
4517                              <1>       ; for(i=0;i<cheight;i++)
4518                              <1>       ; {
4519                              <1>       ;  memsetb(0xa000,dest+i*nbcols,attr,cols);
4520                              <1>       ; }
4521 0000284D 81C700000A00        <1>       add    edi, 0A0000h
4522 00002853 51                  <1>       push   ecx
4523                              <1>       ; 08/08/2016
4524 00002854 8A35[DE6F0000]      <1>       mov    dh, [CHAR_HEIGHT]
4525                              <1>       ;; 07/08/2016
4526                              <1>       ;mov   dh, 8 ; 07/08/2016
4527 0000285A 28D2                <1>       sub    dl, dl ; i
4528                              <1> vgamem_fill_l8_0:
4529 0000285C 57                  <1>       push   edi
4530 0000285D 0FB605[DC6F0000]    <1>       movzx  eax, byte [CRT_COLS]
4531 00002864 F6E2                <1>       mul    dl
4532                              <1>       ; eax = i * nbcols
4533 00002866 66C1E003            <1>       shl    ax, 3 ; * 8 ; 06/08/2016
4534 0000286A 01C7                <1>       add    edi, eax ; dest+i*nbcols
4535 0000286C 88D8                <1>       mov    al, bl ; attr ; 04/08/2016
4536 0000286E 0FB6CF              <1>       movzx  ecx, bh ; cols
4537 00002871 66C1E103            <1>       shl    cx, 3 ; * 8 ; 06/08/2016
4538 00002875 F3AA                <1>       rep    stosb
4539 00002877 5F                  <1>       pop    edi
4540 00002878 FEC2                <1>       inc    dl ; 06/08/2016
4541 0000287A FECE                <1>       dec    dh
4542 0000287C 75DE                <1>       jnz    short vgamem_fill_l8_0
4543                              <1> vgamem_fill_l8_1:
4544 0000287E 59                  <1>       pop    ecx
4545                              <1>
4546                              <1>       ;; outw(VGAREG_GRDC_ADDRESS, 0x0005);
4547                              <1>       ;mov   ax, 0005h
4548                              <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
4549                              <1>       ;out   dx, ax
4550                              <1>
4551 0000287F 58                  <1>       pop    eax
4552 00002880 5A                  <1>       pop    edx
4553                              <1>
4554 00002881 C3                  <1>       retn
4555                              <1>
4556                              <1> vga_graphics_down:
4557                              <1>       ; 08/08/2016
4558                              <1>       ; 07/08/2016
4559                              <1>       ; 31/07/2016
4560                              <1>       ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
4561                              <1>       ;
4562                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
4563                              <1>       ; vgabios-0.7a (2011)
4564                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
4565                              <1>       ; 'vgabios.c', 'biosfn_scroll'
4566                              <1>       ;
4567                              <1>
4568                              <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>
00002882 FC                     <1>         cld     ; !!! Clear direction flag !!!
                                <1>
00002883 88C7                   <1>       mov   bh, al ; 31/07/2016
                                <1>
00002885 BE[F66F0000]           <1>       mov   esi, vga_modes
0000288A 89F7                   <1>       mov   edi, esi
0000288C 83C710                 <1>       add   edi, vga_mode_count
                                <1> vga_g_down_0:
0000288F AC                     <1>       lodsb
00002890 38E0                   <1>       cmp   al, ah ; [CRT_MODE]
00002892 7405                   <1>       je    short vga_g_down_1
00002894 39FE                   <1>       cmp   esi, edi
00002896 72F7                   <1>       jb    short vga_g_down_0
                                <1>       ; xor  bh, bh ; 31/07/2016
00002898 C3                     <1>       retn  ; nothing to do
                                <1> vga_g_down_1:
00002899 88F8                   <1>       mov   al, bh ; 31/07/2016
0000289B 83C64F                 <1>       add   esi, vga_memmodel - (vga_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
                                <1>
                                <1>       ; if(rlr>=nbrows)rlr=nbrows-1;
                                <1>       ; if(clr>=nbcols)clr=nbcols-1;
                                <1>       ; if(nblines>nbrows)nblines=0;
                                <1>       ; cols=clr-cul+1;
                                <1>
0000289E 3A35[E26F0000]         <1>       cmp   dh, [VGA_ROWS]
000028A4 7208                   <1>       jb    short vga_g_down_2
000028A6 8A35[E26F0000]         <1>       mov   dh, [VGA_ROWS]
000028AC FECE                   <1>       dec   dh
                                <1> vga_g_down_2:
000028AE 3A15[DC6F0000]         <1>       cmp   dl, [CRT_COLS]  ; = [VGA_COLS]
000028B4 7208                   <1>       jb    short vga_g_down_3
000028B6 8A15[DC6F0000]         <1>       mov   dl, [CRT_COLS]
000028BC FECA                   <1>       dec   dl
                                <1> vga_g_down_3:
000028BE 3A05[E26F0000]         <1>       cmp   al, [VGA_ROWS]
000028C4 7602                   <1>       jna   short vga_g_down_4
000028C6 28C0                   <1>       sub   al, al ; 0
                                <1> vga_g_down_4:
000028C8 88F7                   <1>       mov   bh, dh ; clr
000028CA 28CF                   <1>       sub   bh, cl ; cul
000028CC FEC7                   <1>       inc   bh ; cols = clr-cul+1
                                <1>
000028CE 20C0                   <1>       and   al, al ; nblines = 0
000028D0 755B                   <1>       jnz   short vga_g_down_6
000028D2 20ED                   <1>       and   ch, ch ; rul = 0
000028D4 7557                   <1>       jnz   short vga_g_down_6
000028D6 20C9                   <1>       and   cl, cl ; cul = 0
000028D8 7553                   <1>       jnz   short vga_g_down_6
                                <1>
000028DA 6650                   <1>       push  ax
000028DC A0[E26F0000]           <1>       mov   al, [VGA_ROWS]
000028E1 FEC8                   <1>       dec   al
000028E3 38C6                   <1>       cmp   dh, al ; rlr = nbrows-1
000028E5 7544                   <1>       jne   short vga_g_down_5
000028E7 A0[DC6F0000]           <1>         mov     al, [CRT_COLS]  ; = VGA_COLS
000028EC FEC8                   <1>       dec   al
000028EE 38C2                   <1>       cmp   dl, al ; clr = nbcols-1
000028F0 7539                   <1>       jne   short vga_g_down_5
000028F2 6658                   <1>       pop   ax
                                <1>
000028F4 66B80502               <1>       mov   ax, 0205h
000028F8 66BACE03               <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
000028FC 66EF                   <1>       out   dx, ax
000028FE A0[E26F0000]           <1>       mov   al, [VGA_ROWS]
00002903 8A25[DC6F0000]         <1>       mov   ah, [CRT_COLS] ; = [VGA_COLS]
00002909 F6E4                   <1>       mul   ah
0000290B 0FB7D0                 <1>       movzx edx, ax
                                <1>       ; 08/08/2016
0000290E 0FB605[DE6F0000]       <1>       movzx eax, byte [CHAR_HEIGHT]
00002915 F7E2                   <1>       mul   edx
                                <1>       ; eax = byte count
00002917 89C1                   <1>       mov   ecx, eax
                                <1>       ;; 07/08/2016
                                <1>       ;shl   dx, 3 ; * 8 ; * [CHAR_HEIGHT]
                                <1>       ;mov   ecx, edx
00002919 88D8                   <1>       mov   al, bl ; fill value for blanked lines
0000291B BF00000A00             <1>       mov   edi, 0A0000h
00002920 F3AA                   <1>       rep   stosb
                                <1>
00002922 B005                   <1>       mov   al, 5
00002924 66BACE03               <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002928 66EF                   <1>       out   dx, ax ; 0005h
                                <1>
0000292A C3                     <1>       retn
                                <1>
                                <1> vga_g_down_5:
0000292B 6658                   <1>       pop   ax
                                <1>
                                <1> vga_g_down_6:
                                <1>       ; [ESI] = VGA memory model number for current video mode
                                <1>         ;
                                <1>         ; LINEAR8 equ 5
                                <1>         ; PLANAR4 equ 4
                                <1>         ; PLANAR1 equ 3
```

```
                                <1>
0000292D 803E04                 <1>       cmp    byte [esi], PLANAR4
00002930 742C                   <1>       je     short vga_g_down_planar
00002932 803E03                 <1>       cmp    byte [esi], PLANAR1
00002935 7427                   <1>       je     short vga_g_down_planar
                                <1> vga_g_down_linear8:
                                <1>       ; 07/07/2016 (TEMPORARY)
                                <1>       ;
                                <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>
                                <1> vga_g_down_l0:
                                <1>       ;{for(i=rlr;i>=rul;i--)
                                <1>         ; if((i<rul+nblines)||(nblines==0))
00002937 08C0                   <1>       or     al, al
00002939 741C                   <1>       jz     short vga_g_down_l2
0000293B 88C4                   <1>       mov    ah, al
0000293D 00EC                   <1>       add    ah, ch
                                <1>       ;jc    short vga_g_down_l2
0000293F 86EE                   <1>       xchg   ch, dh
00002941 38E5                   <1>       cmp    ch, ah
00002943 7212                   <1>       jb     short vga_g_down_l2
00002945 88EC                   <1>       mov    ah, ch
00002947 28C4                   <1>       sub    ah, al ; ah = i - nblines
                                <1>       ; else
                                <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
00002949 E877FEFFFF             <1>       call   vgamem_copy_l8
                                <1> vga_g_down_l1:
0000294E 86F5                   <1>       xchg   dh, ch
00002950 FECE                   <1>       dec    dh
00002952 38EE                   <1>       cmp    dh, ch
00002954 73E1                   <1>       jnb    short vga_g_down_l0
00002956 C3                     <1>       retn
                                <1>
                                <1> vga_g_down_l2:
                                <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
00002957 E8D3FEFFFF             <1>       call   vgamem_fill_l8
0000295C 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))
0000295E 08C0                   <1>       or     al, al
00002960 741C                   <1>       jz     short vga_g_down_pl2
00002962 88C4                   <1>       mov    ah, al
00002964 00EC                   <1>       add    ah, ch
                                <1>       ;jc    short vga_g_down_pl2
00002966 86EE                   <1>       xchg   ch, dh
00002968 38E5                   <1>       cmp    ch, ah
0000296A 7212                   <1>       jb     short vga_g_down_pl2
0000296C 88EC                   <1>       mov    ah, ch
0000296E 28C4                   <1>       sub    ah, al ; ah = i - nblines
                                <1>       ; else
                                <1>       ; vgamem_copy_pl4(cul,i,i-nblines,cols,nbcols,cheight);
00002970 E88BFDFFFF             <1>       call   vgamem_copy_pl4
                                <1> vga_g_down_pl1:
00002975 86F5                   <1>       xchg   dh, ch
00002977 FECE                   <1>       dec    dh
00002979 38EE                   <1>       cmp    dh, ch
0000297B 73E1                   <1>       jnb    short vga_g_down_pl0
0000297D C3                     <1>       retn
                                <1>
                                <1> vga_g_down_pl2:
                                <1>       ; vgamem_fill_pl4(cul,i,cols,nbcols,cheight,attr);
0000297E E8EBFDFFFF             <1>       call   vgamem_fill_pl4
00002983 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
```

```
4779 00002985 80FC07              <1>       cmp    ah, 7
4780 00002988 0F87F4FEFFFF        <1>         ja      vga_graphics_down
4781                              <1>       ;je    _n0
4782                              <1>
4783 0000298E 88C7                <1>       MOV    bh, al              ; save line count in BH
4784 00002990 6689D0              <1>       MOV    AX, DX              ; GET LOWER RIGHT POSITION INTO AX REG
4785                              <1>
4786                              <1> ;-----      USE CHARACTER SUBROUTINE FOR POSITIONING
4787                              <1> ;-----      ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
4788                              <1>
4789 00002993 E8F2010000          <1>       CALL   GRAPH_POSN
4790 00002998 0FB7F8              <1>       MOVzx  eDI, AX                  ; SAVE RESULT AS DESTINATION ADDRESS
4791                              <1>
4792                              <1> ;-----      DETERMINE SIZE OF WINDOW
4793                              <1>
4794 0000299B 6629CA              <1>       SUB    DX, CX
4795 0000299E 6681C20101          <1>         ADD     DX, 101h               ; ADJUST VALUES
4796 000029A3 C0E602              <1>       SAL    DH, 2               ; MULTIPLY ROWS BY 4 AT 8 VERT DOTS/CHAR
4797                              <1>                                  ; AND EVEN/ODD ROWS
4798                              <1>
4799                              <1> ;-----      DETERMINE CRT MODE
4800                              <1>
4801 000029A6 803D[DA6F0000]06    <1>       CMP    byte [CRT_MODE], 6  ; TEST FOR MEDIUM RES
4802 000029AD 7307                <1>         JNC     short _R12              ; FIND_SOURCE_DOWN
4803                              <1>
4804                              <1> ;-----      MEDIUM RES DOWN
4805 000029AF D0E2                <1>       SAL    DL, 1               ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
4806 000029B1 66D1E7              <1>       SAL    DI, 1               ; OFFSET *2 SINCE 2 BYTES/CHAR
4807 000029B4 6647                <1>       INC    DI                  ; POINT TO LAST BYTE
4808                              <1>
4809                              <1> ;-----      DETERMINE THE SOURCE ADDRESS IN THE BUFFER
4810                              <1>
4811                              <1>  R12:                                   ; FIND_SOURCE_DOWN
4812 000029B6 81C700800B00        <1>       add    edi, 0B8000h
4813 000029BC 6681C7F000          <1>       ADD    DI, 240                  ; POINT TO LAST ROW OF PIXELS
4814 000029C1 C0E702              <1>       sal    bh, 2               ; multiply number of lines by 4
4815 000029C4 74(06)              <1>       JZ     short 6                  ; IF ZERO, THEN BLANK ENTIRE FIELD
4816 000029C6 B050                <1>       MOV    AL, 80              ; 80 BYTES/ROW
4817 000029C8 F6E7                <1>       mul    bh                  ; determine offset to source
4818 000029CA 89FE                <1>       MOV    eSI, eDI            ; SET UP SOURCE
4819 000029CC 6629C6              <1>       SUB    SI, AX              ; SUBTRACT THE OFFSET
4820 000029CF 88F4                <1>       MOV    AH, DH              ; NUMBER OF ROWS IN FIELD
4821 000029D1 28FC                <1>       sub    ah, bh              ; determine number to move
4822                              <1>
4823                              <1> ;-----      LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
4824                              <1>
4825                              <1> _R13:                                   ; ROW_LOOP_DOWN
4826 000029D3 E824000000          <1>         CALL    _R17                    ; MOVE ONE ROW
4827 000029D8 6681EE5020          <1>       SUB    SI, 2000h+80        ; MOVE TO NEXT ROW
4828 000029DD 6681EF5020          <1>       SUB    DI, 2000h+80
4829 000029E2 FECC                <1>       DEC    AH                  ; NUMBER OF ROWS TO MOVE
4830 000029E4 75ED                <1>         JNZ     short _R13              ; CONTINUE TILL ALL MOVED
4831                              <1>
4832                              <1> ;-----      FILL IN THE VACATED LINE(S)
4833                              <1> _R14:                                   ; CLEAR_ENTRY_DOWN
4834 000029E6 88D8                <1>       mov    al, bl              ; attribute to fill with
4835                              <1> _R15_:                                  ; CLEAR_LOOP_DOWN
4836 000029E8 E82B000000          <1>       CALL   _R18                ; CLEAR A ROW
4837 000029ED 6681EF5020          <1>       SUB    DI, 2000h+80        ; POINT TO NEXT LINE
4838 000029F2 FECF                <1>       dec    bh                  ; number of lines to fill
4839 000029F4 75F2                <1>         JNZ     short _R15_             ; CLEAR_LOOP_DOWN
4840                              <1>       ; 18/11/2020
4841 000029F6 FC                  <1>       CLD                        ; RESET THE DIRECTION FLAG
4842                              <1>
4843 000029F7 C3                  <1>       retn                       ; EVERYTHING DONE
4844                              <1>
4845                              <1> _R16:                                   ; BLANK_FIELD_DOWN
4846 000029F8 88F7                <1>       mov    bh, dh              ; set blank count to everything in field
4847 000029FA EBEA                <1>         JMP     short _R14              ; CLEAR THE FIELD
4848                              <1>
4849                              <1> ; 27/06/2016 - TRDOS 386 (TRDOS v2.0)
4850                              <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
4851                              <1>
4852                              <1> ;-----      ROUTINE TO MOVE ONE ROW OF INFORMATION
4853                              <1>
4854                              <1> _R17:
4855 000029FC 0FB6CA              <1>       MOVzx  ecx, DL                   ; NUMBER OF BYTES IN THE ROW
4856 000029FF 56                  <1>       PUSH   eSI
4857 00002A00 57                  <1>       PUSH   eDI                 ; SAVE POINTERS
4858 00002A01 F3A4                <1>       REP    MOVSB               ; MOVE THE EVEN FIELD
4859 00002A03 5F                  <1>       POP    eDI
4860 00002A04 5E                  <1>       POP    eSI
4861 00002A05 6681C60020          <1>       ADD    SI, 2000h
4862 00002A0A 6681C70020          <1>       ADD    DI, 2000h           ; POINT TO THE ODD FIELD
4863 00002A0F 56                  <1>       PUSH   eSI
4864 00002A10 57                  <1>       PUSH   eDI                 ; SAVE THE POINTERS
4865 00002A11 88D1                <1>       MOV    CL, DL              ; COUNT BACK
4866 00002A13 F3A4                <1>       REP    MOVSB               ; MOVE THE ODD FIELD
4867 00002A15 5F                  <1>       POP    eDI
4868 00002A16 5E                  <1>       POP    eSI                 ; POINTERS BACK
4869 00002A17 C3                  <1>       RETn                       ; RETURN TO CALLER
4870                              <1>
4871                              <1> ;-----      CLEAR A SINGLE ROW
4872                              <1>
4873                              <1> _R18:
4874 00002A18 0FB6CA              <1>       MOVzx  ecx, DL                   ; NUMBER OF BYTES IN FIELD
4875 00002A1B 57                  <1>       PUSH   eDI                 ; SAVE POINTER
4876 00002A1C F3AA                <1>       REP    STOSB               ; STORE THE NEW VALUE
4877 00002A1E 5F                  <1>       POP    eDI                 ; POINTER BACK
4878 00002A1F 6681C70020          <1>       ADD    DI, 2000h           ; POINT TO ODD FIELD
4879 00002A24 57                  <1>       PUSH   eDI
4880 00002A25 88D1                <1>       MOV    CL, DL
4881 00002A27 F3AA                <1>       REP    STOSB               ; FILL THE ODD FIELD
4882 00002A29 5F                  <1>       POP    eDI
4883 00002A2A C3                  <1>       RETn                       ; RETURN TO CALLER
```

```
                                <1>
                                <1> ; 04/07/2016
                                <1> ; 01/07/2016
                                <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
                                <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                                <1> ;-------------------------------------------------
                                <1> ; GRAPHICS WRITE
                                <1> ;  THIS ROUTINE WRITES THE ASCII CHARACTER TO THE CURRENT
                                <1> ;  POSITION ON THE SCREEN.
                                <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:
00002A2B 25FF000000             <1>       and   eax, 0FFh           ; ZERO TO HIGH OF CODE POINT
00002A30 50                     <1>       PUSH  eAX                 ; SAVE CODE POINT VALUE
                                <1>
                                <1> ;-----      DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
                                <1>
00002A31 E84D010000             <1>       CALL  S26                 ; FIND LOCATION IN REGEN BUFFER
00002A36 89C7                   <1>       MOV   eDI, eAX            ; REGEN POINTER IN DI
                                <1>
                                <1> ;-----      DETERMINE REGION TO GET CODE POINTS FROM
                                <1>
00002A38 58                     <1>       POP   eAX                 ; RECOVER CODE POINT
                                <1>
00002A39 BE[A8600100]           <1>       MOV   eSI, CRT_CHAR_GEN   ; OFFSET OF IMAGES
                                <1>
                                <1> ;-----      DETERMINE GRAPHICS MODE IN OPERATION
                                <1>                                 ; DETERMINE_MODE
00002A3E 66C1E003               <1>       SAL   AX, 3               ; MULTIPLY CODE POINT VALUE BY 8
00002A42 01C6                   <1>       ADD   eSI, eAX            ; SI HAS OFFSET OF DESIRED CODES
                                <1>
00002A44 803D[DA6F0000]06       <1>       CMP   byte [CRT_MODE], 6
00002A4B 7231                   <1>       JC    short S6            ; TEST FOR MEDIUM RESOLUTION MODE
                                <1>
                                <1> ;-----      HIGH RESOLUTION MODE
                                <1>
00002A4D 81C700800B00           <1>       add   edi, 0B8000h
                                <1> S1:                             ; HIGH_CHAR
00002A53 57                     <1>       PUSH  eDI                 ; SAVE REGEN POINTER
00002A54 56                     <1>       PUSH  eSI                 ; SAVE CODE POINTER
00002A55 B604                   <1>       MOV   DH, 4               ; NUMBER OF TIMES THROUGH LOOP
                                <1> S2:
00002A57 AC                     <1>       LODSB                     ; GET BYTE FROM CODE POINTS
00002A58 F6C380                 <1>       TEST  BL, 80H                   ; SHOULD WE USE THE FUNCTION
00002A5B 7515                   <1>       JNZ   short S5            ; TO PUT CHAR IN
00002A5D AA                     <1>       STOSB                     ; STORE IN REGEN BUFFER
00002A5E AC                     <1>       LODSB
                                <1> S4:
00002A5F 8887FF1F0000           <1>       MOV   [eDI+2000H-1], AL ; STORE IN SECOND HALF
00002A65 83C74F                 <1>       ADD   eDI, 79                   ; MOVE TO NEXT ROW IN REGEN
00002A68 FECE                   <1>       DEC   DH                  ; DONE WITH LOOP
00002A6A 75EB                   <1>       JNZ   short S2
00002A6C 5E                     <1>       POP   eSI
00002A6D 5F                     <1>       POP   eDI                 ; RECOVER REGEN POINTER
00002A6E 47                     <1>       INC   eDI                 ; POINT TO NEXT CHAR POSITION
00002A6F E2E2                   <1>       LOOP  S1                  ; MORE CHARS TO WRITE
00002A71 C3                     <1>       retn
                                <1>
                                <1> S5:
00002A72 3207                   <1>       XOR   AL, [eDI]           ; EXCLUSIVE OR WITH CURRENT
00002A74 AA                     <1>       STOSB                     ; STORE THE CODE POINT
00002A75 AC                     <1>       LODSB                     ; AGAIN FOR ODD FIELD
00002A76 3287FF1F0000           <1>       XOR   AL, [eDI+2000H-1]
00002A7C EBE1                   <1>       JMP   short S4            ; BACK TO MAINSTREAM
                                <1>
                                <1> ;-----      MEDIUM RESOLUTION WRITE
                                <1> S6:                             ; MED_RES_WRITE
00002A7E 88DA                   <1>       MOV   DL, BL              ; SAVE HIGH COLOR BIT
00002A80 66D1E7                 <1>       SAL   DI, 1               ; OFFSET*2 SINCE 2 BYTES/CHAR
                                <1>                                 ; EXPAND BL TO FULL WORD OF COLOR
00002A83 80E303                 <1>       AND   BL, 3               ; ISOLATE THE COLOR BITS ( LOW 2 BITS )
00002A86 B055                   <1>       MOV   AL, 055H            ; GET BIT CONVERSION MULTIPLIER
00002A88 F6E3                   <1>       MUL   BL                  ; EXPAND 2 COLOR BITS TO 4 REPLICATIONS
00002A8A 88C3                   <1>       MOV   BL, AL              ; PLACE BACK IN WORK REGISTER
00002A8C 88C7                   <1>       MOV   BH, AL              ; EXPAND TO 8 REPLICATIONS OF COLOR BITS
00002A8E 81C700800B00           <1>       add   edi, 0B8000h
                                <1> S7:                                   ; MED_CHAR
00002A94 57                     <1>       PUSH  eDI                 ; SAVE REGEN POINTER
00002A95 56                     <1>       PUSH  eSI                 ; SAVE THE CODE POINTER
00002A96 B604                   <1>       MOV   DH, 4               ; NUMBER OF LOOPS
```

```
4989                            <1> S8:
4990 00002A98 AC                <1>         LODSB                   ; GET CODE POINT
4991 00002A99 E8B3000000        <1>         CALL    S21             ; DOUBLE UP ALL THE BITS
4992 00002A9E 6621D8            <1>         AND     AX, BX          ; CONVERT TO FOREGROUND COLOR ( 0 BACK )
4993 00002AA1 86E0              <1>         XCHG    AH, AL          ; SWAP HIGH/LOW BYTES FOR WORD MOVE
4994 00002AA3 F6C280            <1>         TEST    DL, 80H                 ; IS THIS XOR FUNCTION
4995 00002AA6 7403              <1>         JZ      short S9        ; NO, STORE IT IN AS IS
4996 00002AA8 663307            <1>         XOR     AX, [eDI]       ; DO FUNCTION WITH LOW/HIGH
4997                            <1> S9:
4998 00002AAB 668907            <1>         MOV     [eDI], AX       ; STORE FIRST BYTE HIGH, SECOND LOW
4999 00002AAE AC                <1>         LODSB                   ; GET CODE POINT
5000 00002AAF E89D000000        <1>         CALL    S21
5001 00002AB4 6621D8            <1>         AND     AX, BX          ; CONVERT TO COLOR
5002 00002AB7 86E0              <1>         XCHG    AH, AL          ; SWAP HIGH/LOW BYTES FOR WORD MOVE
5003 00002AB9 F6C280            <1>         TEST    DL, 80H                 ; AGAIN, IS THIS XOR FUNCTION
5004 00002ABC 7407              <1>         JZ      short _S10      ; NO, JUST STORE THE VALUES
5005 00002ABE 66338700200000    <1>         XOR     AX, [eDI+2000H]         ; FUNCTION WITH FIRST HALF LOW
5006                            <1> _S10:
5007 00002AC5 66898700200000    <1>         MOV     [eDI+2000H], AX         ; STORE SECOND PORTION HIGH
5008 00002ACC 6683C750          <1>         ADD     DI, 80          ; POINT TO NEXT LOCATION
5009 00002AD0 FECE              <1>         DEC     DH
5010 00002AD2 75C4              <1>         JNZ     short S8        ; KEEP GOING
5011 00002AD4 5E                <1>         POP     eSI             ; RECOVER CODE POINTER
5012 00002AD5 5F                <1>         POP     eDI             ; RECOVER REGEN POINTER
5013 00002AD6 47                <1>         INC     eDI             ; POINT TO NEXT CHAR POSITION
5014 00002AD7 47                <1>         INC     eDI
5015 00002AD8 E2BA              <1>         LOOP    S7              ; MORE TO WRITE
5016 00002ADA C3                <1>         retn
5017                            <1>
5018                            <1> ; 04/07/2016
5019                            <1> ; 01/07/2016
5020                            <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
5021                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5022                            <1> ;----------------------------------------
5023                            <1> ; GRAPHICS READ
5024                            <1> ;----------------------------------------
5025                            <1> GRAPHICS_READ:
5026 00002ADB E8A3000000        <1>         CALL    S26             ; CONVERTED TO OFFSET IN REGEN
5027 00002AE0 89C6              <1>         MOV     eSI, eAX        ; SAVE IN SI
5028 00002AE2 81C600800B00      <1>         add     esi, 0B8000h    ; 01/07/2016
5029 00002AE8 83EC08            <1>         SUB     eSP, 8          ; ALLOCATE SPACE FOR THE READ CODE POINT
5030 00002AEB 89E5              <1>         MOV     eBP, eSP        ; POINTER TO SAVE AREA
5031                            <1>
5032                            <1> ;-----      DETERMINE GRAPHICS MODES
5033 00002AED B604              <1>         mov     dh, 4           ; number of passes ; 01/07/2016
5034 00002AEF 803D[DA6F0000]06  <1>         CMP     byte [CRT_MODE], 6
5035 00002AF6 7219              <1>         JC      short S12       ; MEDIUM RESOLUTION
5036                            <1>
5037                            <1> ;-----      HIGH RESOLUTION READ
5038                            <1> ;-----      GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
5039                            <1>         ;MOV    DH,4            ; NUMBER OF PASSES
5040                            <1> S11:
5041 00002AF8 8A06              <1>         MOV     AL, [eSI]       ; GET FIRST BYTE
5042 00002AFA 884500            <1>         MOV     [eBP], AL       ; SAVE IN STORAGE AREA
5043 00002AFD 45                <1>         INC     eBP             ; NEXT LOCATION
5044 00002AFE 8A8600200000      <1>         MOV     AL, [eSI+2000H]         ; GET LOWER REGION BYTE
5045 00002B04 884500            <1>         MOV     [eBP], AL       ; ADJUST AND STORE
5046 00002B07 45                <1>         INC     eBP
5047 00002B08 83C650            <1>         ADD     eSI, 80                 ; POINTER INTO REGEN
5048 00002B0B FECE              <1>         DEC     DH              ; LOOP CONTROL
5049 00002B0D 75E9              <1>         JNZ     short S11       ; DO IT SOME MORE
5050 00002B0F EB1D              <1>         JMP     SHORT S14       ; GO MATCH THE SAVED CODE POINTS
5051                            <1>
5052                            <1> ;-----      MEDIUM RESOLUTION READ
5053                            <1> S12:
5054 00002B11 66D1E6            <1>         SAL     SI, 1           ; OFFSET*2 SINCE 2 BYTES/CHAR
5055                            <1>         ;MOV    DH, 4           ; NUMBER OF PASSES
5056                            <1> S13:
5057 00002B14 E84D000000        <1>         CALL    S23             ; GET BYTES FROM REGEN INTO SINGLE SAVE
5058 00002B19 81C6FE1F0000      <1>         ADD     eSI, 2000H-2    ; GO TO LOWER REGION
5059 00002B1F E842000000        <1>         CALL    S23             ; GET THIS PAIR INTO SAVE
5060 00002B24 81EEB21F0000      <1>         SUB     eSI, 2000H-80+2         ; ADJUST POINTER BACK INTO UPPER
5061 00002B2A FECE              <1>         DEC     DH
5062 00002B2C 75E6              <1>         JNZ     short S13       ; KEEP GOING UNTIL ALL 8 DONE
5063                            <1>
5064                            <1> ;-----      SAVE AREA HAS CHARACTER IN IT, MATCH IT
5065                            <1> S14:                            ; FIND_CHAR
5066 00002B2E BF[A8600100]      <1>         MOV     eDI, CRT_CHAR_GEN ; ESTABLISH ADDRESSING
5067 00002B33 83ED08            <1>         SUB     eBP, 8          ; ADJUST POINTER TO START OF SAVE AREA
5068 00002B36 89EE              <1>         MOV     eSI, eBP
5069                            <1> S15:
5070 00002B38 66B80001          <1>         mov     ax, 256                 ; NUMBER TO TEST AGAINST
5071                            <1> S16:
5072 00002B3C 56                <1>         PUSH    eSI             ; SAVE SAVE AREA POINTER
5073 00002B3D 57                <1>         PUSH    eDI             ; SAVE CODE POINTER
5074                            <1>         ;MOV    eCX, 4          ; NUMBER OF WORDS TO MATCH
5075                            <1>         ;REPE   CMPSW           ; COMPARE THE 8 BYTES AS WORDS
5076 00002B3E A7                <1>         cmpsd                   ; compare first 4 bytes
5077 00002B3F 7501              <1>         jne     short S17       ;
5078 00002B41 A7                <1>         cmpsd                   ; compare last 4 bytes
5079                            <1> S17:
5080 00002B42 5F                <1>         POP     eDI             ; RECOVER THE POINTERS
5081 00002B43 5E                <1>         POP     eSI
5082                            <1>         ;JZ     short S18       ; IF ZERO FLAG SET, THEN MATCH OCCURRED
5083 00002B44 7407              <1>         je      short S18
5084                            <1>         ;                       ; NO MATCH, MOVE ON TO NEXT
5085 00002B46 83C708            <1>         ADD     eDI, 8          ; NEXT CODE POINT
5086 00002B49 6648              <1>         dec     ax              ; LOOP CONTROL
5087 00002B4B 75EF              <1>         JNZ     short S16       ; DO ALL OF THEM
5088                            <1>
5089                            <1> ;-----      CHARACTER IS FOUND ( AL=0 IF NOT FOUND )
5090                            <1> S18:
5091 00002B4D 83C408            <1>         ADD     eSP, 8          ; READJUST THE STACK, THROW AWAY SAVE
5092 00002B50 C3                <1>         retn                    ; ALL DONE
5093                            <1>
```

```
                        <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                        <1> ;--------------------------------------------
                        <1> ; EXPAND BYTE
                        <1> ;  THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ALL
                        <1> ;  OF THE BITS, TURNING THE 8 BITS INTO 16 BITS.
                        <1> ;  THE RESULT IS LEFT IN AX
                        <1> ;--------------------------------------------
                        <1> S21:
00002B51 6651           <1>       PUSH   CX                ; SAVE REGISTER
                        <1>       ;MOV   CX, 8             ; SHIFT COUNT REGISTER FOR ONE BYTE
00002B53 B108           <1>       mov    cl, 8
                        <1> S22:
00002B55 D0C8           <1>       ROR    AL,1              ; SHIFT BITS, LOW BIT INTO CARRY FLAG
00002B57 66D1DD         <1>       RCR    BP,1              ; MOVE CARRY FLAG (LOW BIT INTO RESULTS
00002B5A 66D1FD         <1>       SAR    BP,1              ; SIGN EXTEND HIGH BIT (DOUBLE IT)
                        <1>       ;LOOP  S22               ; REPEAT FOR ALL 8 BITS
00002B5D FEC9           <1>       dec    cl
00002B5F 75F4           <1>       jnz    short S22
00002B61 6695           <1>       XCHG   AX, BP            ; MOVE RESULTS TO PARAMETER REGISTER
00002B63 6659           <1>       POP    CX                ; RECOVER REGISTER
00002B65 C3             <1>       RETn                     ; ALL DONE
                        <1>
                        <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                        <1> ;--------------------------------------------------
                        <1> ; MED_READ_BYTE
                        <1> ; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN BUFFER,
                        <1> ;  COMPARE AGAINST THE CURRENT FOREGROUND COLOR, AND PLACE
                        <1> ;  THE CORRESPONDING ON/OFF BIT PATTERN INTO THE CURRENT
                        <1> ;  POSITION IN THE SAVE AREA
                        <1> ; ENTRY --
                        <1> ;  SI,DS = POINTER TO REGEN AREA OF INTEREST
                        <1> ;  BX = EXPANDED FOREGROUND COLOR
                        <1> ;  BP = POINTER TO SAVE AREA
                        <1> ; EXIT --
                        <1> ;  SI AND BP ARE INCREMENTED
                        <1> ;--------------------------------------------------
                        <1> S23:
00002B66 66AD           <1>       LODSW                    ; GET FIRST BYTE AND SECOND BYTES
00002B68 86C4           <1>       XCHG   AL, AH            ; SWAP FOR COMPARE
00002B6A 66B900C0       <1>       MOV    CX, 0C000H        ; 2 BIT MASK TO TEST THE ENTRIES
00002B6E B200           <1>       MOV    DL, 0             ; RESULT REGISTER
                        <1> S24:
00002B70 6685C8         <1>       TEST   AX, CX            ; IS THIS SECTION BACKCROUND?
00002B73 7401           <1>         JZ     short S25              ; IF ZERO, IT IS BACKGROUND (CARRY=0)
00002B75 F9             <1>       STC                      ; WASN'T, SO SET CARRY
                        <1> S25:
00002B76 D0D2           <1>       RCL    DL, 1             ; MOVE THAT BIT INTO THE RESULT
00002B78 66C1E902       <1>       SHR    CX, 2             ; MOVE THE MASK TO THE RIGHT BY 2 BITS
00002B7C 73F2           <1>       JNC    short S24         ; DO IT AGAIN IF MASK DIDN'T FALL OUT
00002B7E 885500         <1>       MOV    [eBP], DL         ; STORE RESULT IN SAVE AREA
00002B81 45             <1>       INC    eBP               ; ADJUST POINTER
00002B82 C3             <1>       RETn                     ; ALL DONE
                        <1>
                        <1> ; 30/06/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                        <1> ;--------------------------------------------
                        <1> ; V4_POSITION
                        <1> ;  THIS ROUTINE TAKES THE CURSOR POSITION CONTAINED IN
                        <1> ;  THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET
                        <1> ;  INTO THE REGEN BUFFER, ASSUMING ONE BYTE/CHAR.
                        <1> ;  FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST
                        <1> ;  BE DOUBLED.
                        <1> ; ENTRY -- NO REGISTERS,MEMORY LOCATION @CURSOR_POSN IS USED
                        <1> ; EXIT--
                        <1> ;  AX CONTAINS OFFSET INTO REGEN BUFFER
                        <1> ;--------------------------------------------
                        <1> S26:
00002B83 0FB705[368A0100]   <1>       movzx  eax, word [CURSOR_POSN]   ; GET CURRENT CURSOR
                        <1> GRAPH_POSN:
00002B8A 53             <1>       PUSH   eBX               ; SAVE REGISTER
00002B8B 0FB6D8         <1>       movzx  ebx, al                 ; SAVE A COPY OF CURRENT CURSOR
00002B8E A0[DC6F0000]       <1>       MOV    AL, [CRT_COLS]          ; GET BYTES PER COLUMN
00002B93 F6E4           <1>       MUL    AH                ; MULTIPLY BY ROWS
00002B95 66C1E002       <1>       SHL    AX, 2             ; MULTIPLY * 4 SINCE 4 ROWS/BYTE
00002B99 01D8           <1>       ADD    eAX, eBX          ; DETERMINE OFFSET
00002B9B 5B             <1>       POP    eBX               ; RECOVER POINTER
00002B9C C3             <1>       RETn                     ; ALL DONE
                        <1>
                        <1> ; 09/07/2016
                        <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
                        <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
                        <1> ;--------------------------------------------
                        <1> ; SET_COLOR
                        <1> ;     THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN COLOR,
                        <1> ;     AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION GRAPHICS
                        <1> ; INPUT
                        <1> ;     (BH) HAS COLOR ID
                        <1> ;           IF BH=0, THE BACKGROUND COLOR VALUE IS SET
                        <1> ;                 FROM THE LOW BITS OF BL (0-31)
                        <1> ;           IF BH=1, THE PALETTE SELECTION IS MADE
                        <1> ;                 BASED ON THE LOW BIT OF BL:
                        <1> ;                       0 = GREEN, RED, YELLOW FOR COLORS 1,2,3
                        <1> ;                       1 = BLUE, CYAN, MAGENTA FOR COLORS 1,2,3
                        <1> ;     (BL) HAS THE COLOR VALUE TO BE USED
                        <1> ; OUTPUT
                        <1> ;     THE COLOR SELECTION IS UPDATED
                        <1> ;--------------------------------------------
                        <1> SET_COLOR:
00002B9D 803D[DA6F0000]07   <1>         cmp     byte [CRT_MODE], 7      ; 09/07/2016
00002BA4 0F87B5EFFFFF       <1>       ja     VIDEO_RETURN       ; nothing to do for VGA modes
                        <1>
                        <1>       ;MOV   DX, [ADDR_6845]           ; I/O PORT FOR PALETTE
                        <1>       ;mov   dx, 3D4h
```

```
5199                                    <1>       ;ADD   DX,5                ; OVERSCAN PORT
5200 00002BAA 66BAD903                  <1>       mov    dx, 3D9h
5201 00002BAE A0[DD6F0000]              <1>       MOV    AL, [CRT_PALETTE]   ; GET THE CURRENT PALETTE VALUE
5202 00002BB3 08FF                      <1>       OR     BH, BH              ; IS THIS COLOR 0?
5203 00002BB5 7512                      <1>       JNZ    short M20           ; OUTPUT COLOR 1
5204                                    <1>
5205                                    <1> ;-----      HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
5206                                    <1>
5207 00002BB7 24E0                      <1>       AND    AL, 0E0H            ; TURN OFF LOW 5 BITS OF CURRENT
5208 00002BB9 80E31F                    <1>       AND    BL, 01FH            ; TURN OFF HIGH 3 BITS OF INPUT VALUE
5209 00002BBC 08D8                      <1>       OR     AL, BL              ; PUT VALUE INTO REGISTER
5210                                    <1> M19:                             ; OUTPUT THE PALETTE
5211 00002BBE EE                        <1>       OUT    DX, AL              ; OUTPUT COLOR SELECTION TO 3D9 PORT
5212 00002BBF A2[DD6F0000]              <1>       MOV    [CRT_PALETTE], AL   ; SAVE THE COLOR VALUE
5213 00002BC4 E996EFFFFF                <1>       JMP    VIDEO_RETURN
5214                                    <1>
5215                                    <1> ;-----      HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
5216                                    <1>
5217                                    <1> M20:
5218 00002BC9 24DF                      <1>       AND    AL, 0DFH            ; TURN OFF PALETTE SELECT BIT
5219 00002BCB D0EB                      <1>       SHR    BL, 1               ; TEST THE LOW ORDER BIT OF BL
5220 00002BCD 73EF                      <1>       JNC    short M19           ; ALREADY DONE
5221 00002BCF 0C20                      <1>       OR     AL, 20H                   ; TURN ON PALETTE SELECT BIT
5222 00002BD1 EBEB                      <1>       JMP    short M19           ; GO DO IT
5223                                    <1>
5224                                    <1> ; 09/07/2016
5225                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5226                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5227                                    <1> ;--------------------------------------------
5228                                    <1> ; READ DOT -- WRITE DOT
5229                                    <1> ; THESE ROUTINES WILL WRITE A DOT, OR READ THE
5230                                    <1> ;  DOT AT THE INDICATED LOCATION
5231                                    <1> ; ENTRY --
5232                                    <1> ;   DX = ROW (0-199)      (THE ACTUAL VALUE DEPENDS ON THE MODE)
5233                                    <1> ;   CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED )
5234                                    <1> ;   AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE,
5235                                    <1> ;     REQUIRED FOR WRITE DOT ONLY, RIGHT JUSTIFIED)
5236                                    <1> ;     BIT 7 OF AL = 1 INDICATES XOR THE VALUE INTO THE LOCATION
5237                                    <1> ;   DS = DATA SEGMENT
5238                                    <1> ;   ES = REGEN SEGMENT
5239                                    <1> ;
5240                                    <1> ; EXIT
5241                                    <1> ;     AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY
5242                                    <1> ;--------------------------------------------
5243                                    <1>
5244                                    <1> READ_DOT:
5245                                    <1>       ; 09/07/2016
5246 00002BD3 8A25[DA6F0000]            <1>       mov    ah,  [CRT_MODE]
5247 00002BD9 80FC07                    <1>       cmp    ah, 7 ; 6!?
5248 00002BDC 760A                      <1>       jna    short read_dot_cga
5249                                    <1>
5250 00002BDE E8D7030000                <1>       call   vga_read_pixel
5251                                    <1>       ; al = pixel value
5252 00002BE3 E97CEFFFFF                <1>       jmp    _video_return
5253                                    <1>
5254                                    <1> read_dot_cga:
5255                                    <1>       ;je    VIDEO_RETURN ; 7
5256 00002BE8 80FC04                    <1>       cmp    ah, 4 ; graphics ?
5257 00002BEB 0F826EEFFFFF              <1>       jb     VIDEO_RETURN ; no, text mode, nothing to do
5258                                    <1>
5259 00002BF1 E855000000                <1>       CALL   R3                  ; DETERMINE BYTE POSITION OF DOT
5260 00002BF6 8A06                      <1>       MOV    AL, [eSI]           ; GET THE BYTE
5261 00002BF8 20E0                      <1>       AND    AL, AH              ; MASK OFF THE OTHER BITS IN THE BYTE
5262 00002BFA D2E0                      <1>       SHL    AL, CL              ; LEFT JUSTIFY THE VALUE
5263 00002BFC 88F1                      <1>       MOV    CL, DH              ; GET NUMBER OF BITS IN RESULT
5264 00002BFE D2C0                      <1>       ROL    AL, CL              ; RIGHT JUSTIFY THE RESULT
5265                                    <1>       ;JMP   VIDEO_RETURN        ; RETURN FROM VIDEO I/O
5266 00002C00 0FB6C0                    <1>       movzx  eax, al
5267 00002C03 E95CEFFFFF                <1>       jmp    _video_return
5268                                    <1>
5269                                    <1> ; 09/07/2016
5270                                    <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5271                                    <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5272                                    <1>
5273                                    <1> WRITE_DOT:
5274                                    <1>       ; 09/07/2016
5275 00002C08 8A25[DA6F0000]            <1>       mov    ah, [CRT_MODE]
5276 00002C0E 80FC07                    <1>       cmp    ah, 7 ; 6!?
5277 00002C11 760A                      <1>       jna    short write_dot_cga
5278                                    <1>
5279 00002C13 E811030000                <1>       call   vga_write_pixel
5280 00002C18 E942EFFFFF                <1>       jmp    VIDEO_RETURN
5281                                    <1>
5282                                    <1> write_dot_cga:
5283                                    <1>       ;je    VIDEO_RETURN ; 7
5284 00002C1D 80FC04                    <1>       cmp    ah, 4 ; graphics ?
5285 00002C20 0F8239EFFFFF              <1>       jb     VIDEO_RETURN ; no, text mode, nothing to do
5286                                    <1>
5287                                    <1>       ;PUSH  AX                  ; SAVE DOT VALUE
5288 00002C26 6650                      <1>       PUSH   AX                  ; TWICE
5289 00002C28 E81E000000                <1>       CALL   R3                  ; DETERMINE BYTE POSITION OF THE DOT
5290 00002C2D D2E8                      <1>       SHR    AL, CL              ; SHIFT TO SET UP THE BITS FOR OUTPUT
5291 00002C2F 20E0                      <1>       AND    AL, AH              ; STRIP OFF THE OTHER BITS
5292 00002C31 8A0E                      <1>       MOV    CL, [eSI]           ; GET THE CURRENT BYTE
5293 00002C33 665B                      <1>       POP    BX                  ; RECOVER XOR FLAG
5294 00002C35 F6C380                    <1>       TEST   BL, 80H                   ; IS IT ON
5295 00002C38 750D                      <1>       JNZ    short R2            ; YES, XOR THE DOT
5296 00002C3A F6D4                      <1>       NOT    AH                  ; SET MASK TO REMOVE THE INDICATED BITS
5297 00002C3C 20E1                      <1>       AND    CL, AH
5298 00002C3E 08C8                      <1>       OR     AL, CL              ; OR IN THE NEW VALUE OF THOSE BITS
5299                                    <1> R1:                              ; FINISH_DOT
5300 00002C40 8806                      <1>       MOV    [eSI], AL           ; RESTORE THE BYTE IN MEMORY
5301                                    <1>       ;POP   AX
5302 00002C42 E918EFFFFF                <1>       JMP    VIDEO_RETURN        ; RETURN FROM VIDEO I/O
5303                                    <1> R2:                              ; XOR_DOT
```

```
5304 00002C47 30C8              <1>       XOR    AL, CL              ; EXCLUSIVE OR THE DOTS
5305 00002C49 EBF5              <1>       JMP    short R1            ; FINISH UP THE WRITING
5306                            <1>
5307                            <1> ; 01/07/2016 - TRDOS 386 (TRDOS v2.0)
5308                            <1> ; VIDEO1.ASM - 24/03/1985 (IBM PC-AT BIOS source code)
5309                            <1>
5310                            <1> ;-----------------------------------------------
5311                            <1> ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION OF THE
5312                            <1> ; INDICATED ROW COLUMN VALUE IN GRAPHICS MODE.
5313                            <1> ; ENTRY --
5314                            <1> ;  DX = ROW VALUE (0-199)
5315                            <1> ;  CX = COLUMN VALUE (0-639)
5316                            <1> ; EXIT --
5317                            <1> ;  SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST
5318                            <1> ;  AH = MASK TO STRIP OFF THE BITS OF INTEREST
5319                            <1> ;  CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH
5320                            <1> ;  DH = # BITS IN RESULT
5321                            <1> ;  BX = MODIFIED
5322                            <1> ;-----------------------------------------------
5323                            <1> R3:
5324                            <1>
5325                            <1> ;-----      DETERMINE 1ST BYTE IN INDICATED ROW BY MULTIPLYING ROW VALUE BY 40
5326                            <1> ;-----       ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW )
5327                            <1>
5328 00002C4B 0FB7F0            <1>       movzx  esi, ax                 ; WILL SAVE AL AND AH DURING OPERATION
5329 00002C4E B028              <1>       MOV    AL, 40
5330 00002C50 F6E2              <1>       MUL    DL                  ; AX= ADDRESS OF START OF INDICATED ROW
5331 00002C52 A808              <1>       TEST   AL, 08H             ; TEST FOR EVEN/ODD ROW CALCULATED
5332 00002C54 7404              <1>       JZ     short R4            ; JUMP IF EVEN ROW
5333 00002C56 6605D81F          <1>       ADD    AX, 2000H-40        ; OFFSET TO LOCATION OF ODD ROWS ADJUST
5334                            <1> R4:                               ; EVEN_ROW
5335 00002C5A 6696              <1>       XCHG   SI, AX              ; MOVE POINTER TO (SI) AND RECOVER (AX)
5336 00002C5C 81C600800B00      <1>       add    esi, 0B8000h
5337 00002C62 6689CA            <1>       MOV    DX, CX              ; COLUMN VALUE TO DX
5338                            <1>
5339                            <1> ;-----      DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
5340                            <1>
5341                            <1> ; SET UP THE REGISTERS ACCORDING TO THE MODE
5342                            <1> ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES )
5343                            <1> ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M )
5344                            <1> ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/C0H FOR H/M )
5345                            <1> ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M )
5346                            <1>
5347 00002C65 66BBC002          <1>       MOV    BX, 2C0H
5348 00002C69 66B90203          <1>       MOV    CX, 302H            ; SET PARMS FOR MED RES
5349 00002C6D 803D[DA6F0000]06  <1>       CMP    byte [CRT_MODE], 6
5350 00002C74 7208              <1>       JC     short R5            ; HANDLE IF MED RES
5351 00002C76 66BB8001          <1>       MOV    BX, 180H
5352 00002C7A 66B90307          <1>       MOV    CX, 703H            ; SET PARMS FOR HIGH RES
5353                            <1>
5354                            <1> ;-----      DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
5355                            <1> R5:
5356 00002C7E 20D5              <1>       AND    CH, DL              ; ADDRESS OF PEL WITHIN BYTE TO CH
5357                            <1>
5358                            <1> ;-----      DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
5359                            <1>
5360 00002C80 66D3EA            <1>       SHR    DX, CL              ; SHIFT BY CORRECT AMOUNT
5361 00002C83 6601D6            <1>       ADD    SI, DX              ; INCREMENT THE POINTER
5362 00002C86 88FE              <1>       MOV    DH, BH              ; GET THE # OF BITS IN RESULT TO DH
5363                            <1>
5364                            <1> ;-----      MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
5365                            <1>
5366 00002C88 28C9              <1>       SUB    CL, CL              ; ZERO INTO STORAGE LOCATION
5367                            <1> R6:
5368 00002C8A D0C8              <1>       ROR    AL, 1               ; LEFT JUSTIFY VALUE IN AL (FOR WRITE)
5369 00002C8C 00E9              <1>       ADD    CL, CH              ; ADD IN THE BIT OFFSET VALUE
5370 00002C8E FECF              <1>       DEC    BH                  ; LOOP CONTROL
5371 00002C90 75F8              <1>       JNZ    short R6            ; ON EXIT, CL HAS COUNT TO RESTORE BITS
5372 00002C92 88DC              <1>       MOV    AH, BL              ;  GET MASK TO AH
5373 00002C94 D2EC              <1>       SHR    AH, CL              ;  MOVE THE MASK TO CORRECT LOCATION
5374 00002C96 C3                <1>       RETn                       ;  RETURN WITH EVERYTHING SET UP
5375                            <1>
5376                            <1> load_dac_palette:
5377                            <1>       ; 29/07/2016
5378                            <1>       ; 23/07/2016
5379                            <1>       ; 03/07/2016 (TRDOS 386 = TRDOS v2.0)
5380                            <1>       ; (set_mode_vga)
5381                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
5382                            <1>       ; vgabios-0.7a (2011)
5383                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
5384                            <1>       ; 'vgabios.c', 'load_dac_palette'
5385                            <1>       ;
5386                            <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
5387                            <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
5388                            <1>       ;
5389                            <1>       ; INPUT -> AH = DAC selection number (3, 2 or 1)
5390                            <1>       ; OUTPUT -> ECX = 0, AX = 0
5391                            <1>       ; (Modifed registers: EAX, ECX, EDX, ESI)
5392                            <1>       ;
5393 00002C97 66BAC803          <1>       mov    dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
5394 00002C9B 28C0              <1>       sub    al, al ; 0
5395 00002C9D EE                <1>       out    dx, al ; 0 ; color index, always 0 at the beginning
5396 00002C9E 6642              <1>       inc    dx   ; 3C9h ; VGAREG_DAC_DATA
5397 00002CA0 B900010000        <1>       mov    ecx, 256   ; always 256*3 values
5398                            <1>       ;push  esi
5399 00002CA5 88E0              <1>       mov    al, ah
5400 00002CA7 B43F              <1>       mov    ah, 3Fh      ; 3Fh except DAC selection number 3
5401 00002CA9 3C02              <1>       cmp    al, 2
5402 00002CAB 7414              <1>       je     short l_dac_p_2
5403 00002CAD 7719              <1>       ja     short l_dac_p_3
5404 00002CAF 20C0              <1>       and    al, al
5405 00002CB1 7507              <1>       jnz    short l_dac_p_1
5406                            <1> l_dac_p_0:
5407 00002CB3 BE[685B0100]      <1>       mov    esi, palette0
5408 00002CB8 EB15              <1>       jmp    short l_dac_p_4
```

```
                                <1> l_dac_p_1:
00002CBA BE[285C0100]           <1>         mov     esi, palette1
00002CBF EB0E                   <1>         jmp     short l_dac_p_4
                                <1> l_dac_p_2:
00002CC1 BE[E85C0100]           <1>         mov     esi, palette2
00002CC6 EB07                   <1>         jmp     short l_dac_p_4
                                <1> l_dac_p_3:
00002CC8 B4FF                   <1>         mov     ah, 0FFh ; dac registers
00002CCA BE[A85D0100]           <1>         mov     esi, palette3
                                <1> l_dac_p_4:
00002CCF AC                     <1>         lodsb
00002CD0 EE                     <1>         out     dx, al  ; Red
00002CD1 AC                     <1>         lodsb
00002CD2 EE                     <1>         out     dx, al ; Green
00002CD3 AC                     <1>         lodsb
00002CD4 EE                     <1>         out     dx, al ; Blue
00002CD5 20E4                   <1>         and     ah, ah
00002CD7 7405                   <1>         jz      short l_dac_p_5
00002CD9 FECC                   <1>         dec     ah
00002CDB E2F2                   <1>         loop    l_dac_p_4
                                <1>         ;pop    esi
00002CDD C3                     <1>         retn
                                <1> l_dac_p_5:
                                <1>         ; 29/07/2016
00002CDE FEC9                   <1>         dec     cl
00002CE0 7407                   <1>         jz      short l_dac_p_7
                                <1>         ;
00002CE2 28C0                   <1>         sub     al, al ; 0
                                <1> l_dac_p_6:
00002CE4 EE                     <1>         out     dx, al ; outb(VGAREG_DAC_DATA,0);
00002CE5 EE                     <1>         out     dx, al
00002CE6 EE                     <1>         out     dx, al
00002CE7 E2FB                   <1>         loop    l_dac_p_6
                                <1> l_dac_p_7:
                                <1>         ;pop    esi
00002CE9 C3                     <1>         retn
                                <1>
                                <1> gray_scale_summing:
                                <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>
00002CEA 66BADA03               <1>         mov     dx, 3DAh ; VGAREG_ACTL_RESET
00002CEE EC                     <1>         in      al, dx
00002CEF 30C0                   <1>         xor     al, al ; 0
00002CF1 66BAC003               <1>         mov     dx, 3C0h ; VGAREG_ACTL_ADDRESS
00002CF5 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:
00002CF6 66BAC703               <1>         mov     dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
00002CFA 88D8                   <1>         mov     al, bl
00002CFC 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;
00002CFD 66BAC903               <1>         mov     dx, 3C9h ; VGAREG_DAC_DATA
00002D01 EC                     <1>         in      al, dx ; red
00002D02 B44D                   <1>         mov     ah, 77 ; 0.3* Red
00002D04 F6E4                   <1>         mul     ah
00002D06 6650                   <1>         push    ax
00002D08 EC                     <1>         in      al, dx ; green
00002D09 B497                   <1>         mov     ah, 151  ; 0.59 * Green
00002D0B F6E4                   <1>         mul     ah
00002D0D 6650                   <1>         push    ax
00002D0F EC                     <1>         in      al, dx ; blue
00002D10 B41C                   <1>         mov     ah, 28 ; 0.11 * Blue
00002D12 F6E4                   <1>         mul     ah
00002D14 665A                   <1>         pop     dx
00002D16 6601D0                 <1>         add     ax, dx
00002D19 665A                   <1>         pop     dx
00002D1B 6601D0                 <1>         add     ax, dx
00002D1E 66058000               <1>         add     ax, 80h
00002D22 B03F                   <1>         mov     al, 3Fh
00002D24 38C4                   <1>         cmp     ah, al ; if(i>0x3f)i=0x3f
00002D26 7602                   <1>         jna     short g_s_s_2
00002D28 88C4                   <1>         mov     ah, al
                                <1> g_s_s_2:
00002D2A 66BAC803               <1>         mov     dx, 3C8h  ; VGAREG_DAC_WRITE_ADDRESS
00002D2E 88D8                   <1>         mov     al, bl ; color index
00002D30 EE                     <1>         out     dx, al
00002D31 88E0                   <1>         mov     al, ah ; intensity
00002D33 6642                   <1>         inc     dx ; 3C9h ; VGAREG_DAC_DATA
00002D35 EE                     <1>         out     dx, al ; R (R=G=B)
00002D36 88E0                   <1>         mov     al, ah ; intensity
00002D38 EE                     <1>         out     dx, al ; G (R=G=B)
00002D39 88E0                   <1>         mov     al, ah ; intensity
00002D3B EE                     <1>         out     dx, al ; B (R=G=B)
00002D3C 6649                   <1>         dec     cx
00002D3E 7404                   <1>         jz      short g_s_s_3
00002D40 FEC3                   <1>         inc     bl    ; next color index value
00002D42 EBB2                   <1>         jmp     short g_s_s_1
                                <1> g_s_s_3:
```

```
5514 00002D44 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
5515 00002D48 EC                  <1>       in     al, dx
5516 00002D49 B020                <1>       mov    al, 20h
5517 00002D4B 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
5518 00002D4F EE                  <1>       out    dx, al ; 20h -> set bit 5
5519                              <1>                       ; (after loading palette regs)
5520 00002D50 C3                  <1>       retn
5521                              <1>
5522                              <1> vga_write_char_attr:
5523                              <1> vga_write_char_only:
5524                              <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
5525                              <1>       ;
5526                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
5527                              <1>       ; vgabios-0.7a (2011)
5528                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
5529                              <1>       ; 'vgabios.c', 'biosfn_write_char_attr'
5530                              <1>       ; 'biosfn_write_char_only'
5531                              <1>
5532                              <1>       ; INPUT ->
5533                              <1>       ; [CRT_MODE] = current video mode (>7)
5534                              <1>       ; CX = Count of characters to write
5535                              <1>       ; AL = Character to write
5536                              <1>       ; BL = Color of character
5537                              <1>       ; OUTPUT ->
5538                              <1>       ; Regen buffer updated
5539                              <1>
5540 00002D51 8A25[DA6F0000]      <1>       mov    ah, [CRT_MODE]
5541 00002D57 668B15[368A0100]    <1>       mov    dx, [CURSOR_POSN] ; cursor pos for page 0
5542                              <1>
5543 00002D5E BE[F66F0000]        <1>       mov    esi, vga_modes
5544 00002D63 89F7                <1>       mov    edi, esi
5545 00002D65 83C710              <1>       add    edi, vga_mode_count
5546                              <1> vga_wca_0:
5547 00002D68 AC                  <1>       lodsb
5548 00002D69 38E0                <1>       cmp    al, ah ; [CRT_MODE]
5549 00002D6B 7405                <1>       je     short vga_wca_2
5550 00002D6D 39FE                <1>       cmp    esi, edi
5551 00002D6F 72F7                <1>       jb     short vga_wca_0
5552                              <1> vga_wca_1:
5553 00002D71 C3                  <1>       retn   ; nothing to do
5554                              <1> vga_wca_2:
5555 00002D72 83C64F              <1>       add    esi, vga_memmodel - (vga_modes + 1)
5556                              <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
5557                              <1>
5558                              <1>       ; biosfn_write_char_attr (car,page,attr,count)
5559                              <1>       ; AL = car, page = 0, BL = attr, CX = count
5560 00002D75 803E04              <1>       cmp    byte [esi], PLANAR4
5561 00002D78 741D                <1>       je     short vga_wca_planar
5562 00002D7A 803E03              <1>       cmp    byte [esi], PLANAR1
5563 00002D7D 7418                <1>       je     short vga_wca_planar
5564                              <1> vga_wca_linear8:
5565                              <1>       ; while((count-->0) && (xcurs<nbcols))
5566                              <1>       ; CX = count
5567 00002D7F 6621C9              <1>       and    cx, cx
5568 00002D82 74ED                <1>       jz     short vga_wca_1
5569 00002D84 3A15[DC6F0000]      <1>       cmp    dl, [CRT_COLS]
5570 00002D8A 73E5                <1>       jnb    short vga_wca_1
5571                              <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
5572                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
5573                              <1>       ; [CRT_COLS] = nbcols
5574 00002D8C E81E000000          <1>       call   write_gfx_char_lin
5575 00002D91 6649                <1>       dec    cx ; count
5576 00002D93 FEC2                <1>       inc    dl ; xcurs
5577 00002D95 EBE8                <1>       jmp    short vga_wca_linear8
5578                              <1> vga_wca_planar:
5579                              <1>       ; while((count-->0) && (xcurs<nbcols))
5580                              <1>       ; CX = count
5581 00002D97 6621C9              <1>       and    cx, cx
5582 00002D9A 74D5                <1>       jz     short vga_wca_1
5583 00002D9C 3A15[DC6F0000]      <1>       cmp    dl, [CRT_COLS]
5584 00002DA2 73CD                <1>       jnb    short vga_wca_1
5585                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
5586                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
5587                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
5588 00002DA4 E8A9000000          <1>       call   write_gfx_char_pl4
5589 00002DA9 6649                <1>       dec    cx ; count
5590 00002DAB FEC2                <1>       inc    dl ; xcurs
5591 00002DAD EBE8                <1>       jmp    short vga_wca_planar
5592                              <1>
5593                              <1> write_gfx_char_lin:
5594                              <1>       ; 08/01/2021
5595                              <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
5596                              <1>       ; 08/08/2016
5597                              <1>       ; 31/07/2016
5598                              <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
5599                              <1>       ;
5600                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
5601                              <1>       ; vgabios-0.7a (2011)
5602                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
5603                              <1>       ; 'vgabios.c', 'write_gfx_char_lin'
5604                              <1>
5605                              <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols)
5606                              <1>       ; INPUT ->
5607                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
5608                              <1>       ; [CRT_COLS] = nbcols
5609                              <1>       ; OUTPUT ->
5610                              <1>       ; Regen buffer updated
5611                              <1>
5612 00002DAF 51                  <1>       push   ecx
5613 00002DB0 53                  <1>       push   ebx
5614 00002DB1 52                  <1>       push   edx
5615 00002DB2 50                  <1>       push   eax
5616                              <1>       ; addr=xcurs*8+ycurs*nbcols*64;
5617                              <1>       ; 08/08/2016
5618 00002DB3 0FB6F0              <1>       movzx  esi, al ; car
```

```
                                <1>       ; 08/01/2021
                                <1>       ;movzx eax, dh ; ycurs
                                <1>       ;mov   ah, [CRT_COLS] ; nbcols
                                <1>       ;mul   ah
00002DB6 A0[DC6F0000]           <1>       mov    al, [CRT_COLS]
00002DBB F6E6                   <1>       mul    dh
                                <1>       ;shl   ax, 6 ; * 64
00002DBD 66C1E003               <1>       shl    ax, 3 ; 8 * ycurs * [CRT_COLS]
                                <1>       ;sub   dh, dh
                                <1>       ;shl   dx, 3 ; xcurs * 8
                                <1>       ;movzx edi, dx
00002DC1 BF00000A00             <1>       mov    edi, 0A0000h
00002DC6 30F6                   <1>       xor    dh, dh
00002DC8 6689D7                 <1>       mov    di, dx
                                <1>       ;movzx edi, dl
00002DCB 66C1E703               <1>       shl    di, 3 ; xcurs * 8
                                <1>       ;xor   dh, dh
00002DCF 8A15[DE6F0000]         <1>       mov    dl, [CHAR_HEIGHT]
00002DD5 66F7E2                 <1>       mul    dx
                                <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
                                <1>       ;add   edi, eax ; addr
                                <1>       ;add   edi, 0A0000h
00002DD8 6601C7                 <1>       add    di, ax
                                <1>       ;shl   si, 3 ; car * 8
00002DDB 30E4                   <1>       xor    ah, ah
00002DDD A0[DE6F0000]           <1>       mov    al, [CHAR_HEIGHT]
00002DE2 66F7E6                 <1>       mul    si
00002DE5 6689C6                 <1>       mov    si, ax
                                <1>       ;; esi = src = car * 8
                                <1>       ; esi = src = car * [CHAR_HEIGHT]
                                <1>       ; i = 0
                                <1>       ;add   esi, vgafont8 ; fdata [src+i]
                                <1>       ; 08/08/2016
00002DE8 A1[C2960100]           <1>       mov    eax, [VGA_INT43H]
00002DED 09C0                   <1>       or     eax, eax ; 0 ?
00002DEF 743F                   <1>       jz     short wfxl_7 ; yes, default font
                                <1>       ;cmp   eax, vgafont16
                                <1>         ;je  short wgfxl_0
                                <1>       ;cmp   eax, vgafont14
                                <1>       ;je    short wgfxl_0
                                <1>       ;cmp   eax, vgafont8
                                <1>       ;je    short wgfxl_0
                                <1>       ;; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ;; user font
                                <1>       ;mov   eax, VGAFONTUSR ; 8x16 or 8x8 or 8x14 font
                                <1>       ;                      ; (256 characters)
                                <1> wgfxl_0:
00002DF1 01C6                   <1>       add    esi, eax
                                <1> wgfxl_1:
00002DF3 28FF                   <1>       sub    bh, bh ; i = 0
                                <1> wgfxl_2:
                                <1>       ; for(i=0;i<8;i++)
00002DF5 57                     <1>       push   edi ; addr
00002DF6 0FB605[DC6F0000]       <1>       movzx  eax, byte [CRT_COLS] ; nbcols
00002DFD F6E7                   <1>       mul    bh ; nbcols*i
00002DFF 66C1E003               <1>       shl    ax, 3 ; i*nbcols*8
                                <1>       ; dest=addr+i*nbcols*8;
00002E03 01C7                   <1>       add    edi, eax ; dest + j ; j = 0
00002E05 B180                   <1>       mov    cl, 80h ; mask = 0x80;
                                <1>       ; esi = fdata + src + i
                                <1>       ; for(j=0;j<8;j++)
00002E07 29D2                   <1>       sub    edx, edx ; j = 0
                                <1> wgfxl_3:
00002E09 8A06                   <1>       mov    al, [esi] ; al = fdata[src+i]
00002E0B 20C8                   <1>       and    al, cl ; if (fdata[src+i] & mask)
00002E0D 7402                   <1>       jz     short wgfxl_4  ; data = 0, zf = 1
00002E0F 88D8                   <1>       mov    al, bl ; data = attr;
                                <1> wgfxl_4:
                                <1>       ; write_byte(0xa000,dest+j,data);
00002E11 AA                     <1>       stosb  ; dest + j (+ 0A0000h)
                                <1>       ;inc   dl ; j++
                                <1>       ;cmp   dl, 8
00002E12 80FA07                 <1>       cmp    dl, 7
00002E15 720E                   <1>       jb     short wgfxl_5
00002E17 5F                     <1>       pop    edi
                                <1>       ; 08/08/2016
                                <1>       ;cmp   bh, 7
                                <1>       ;jnb   short wgfxl_6
00002E18 FEC7                   <1>       inc    bh ; i++
00002E1A 3A3D[DE6F0000]         <1>       cmp    bh, [CHAR_HEIGHT]
00002E20 7309                   <1>       jnb    short wgfxl_6
00002E22 46                     <1>       inc    esi
00002E23 EBD0                   <1>       jmp    short wgfxl_2
                                <1> wgfxl_5:
00002E25 D0E9                   <1>       shr    cl, 1 ; mask >>= 1;
00002E27 FEC2                   <1>       inc    dl ; j++
00002E29 EBDE                   <1>         jmp     short wgfxl_3
                                <1> wgfxl_6:
00002E2B 58                     <1>       pop    eax
00002E2C 5A                     <1>       pop    edx
00002E2D 5B                     <1>       pop    ebx
00002E2E 59                     <1>       pop    ecx
00002E2F C3                     <1>       retn
                                <1> wfxl_7:
                                <1>       ; 08/01/2021
                                <1>       ; 05/01/2021
                                <1>       ; Default font (8x8 or 8x14 or 8x16)
00002E30 A0[DE6F0000]           <1>       mov    al, [CHAR_HEIGHT]
00002E35 3C08                   <1>       cmp    al, 8
00002E37 7507                   <1>       jne    short wfxl_8
00002E39 B8[A8600100]           <1>       mov    eax, vgafont8
00002E3E EBB1                   <1>       jmp    short wgfxl_0
                                <1> wfxl_8:
00002E40 3C0E                   <1>       cmp    al, 14
00002E42 7507                   <1>       jne    short wfxl_9
```

```
5724 00002E44 B8[A8680100]        <1>       mov    eax, vgafont14
5725 00002E49 EBA6                <1>       jmp    short wgfxl_0
5726                              <1> wfxl_9:
5727 00002E4B B8[A8760100]        <1>       mov    eax, vgafont16
5728 00002E50 EB9F                <1>       jmp    short wgfxl_0
5729                              <1>
5730                              <1> write_gfx_char_pl4:
5731                              <1>       ; 08/08/2016
5732                              <1>       ; 08/07/2016 (TRDOS 386 = TRDOS v2.0)
5733                              <1>       ;
5734                              <1>       ; derived from 'Plex86/Bochs VGABios' source code
5735                              <1>       ; vgabios-0.7a (2011)
5736                              <1>       ; by the LGPL VGABios developers Team (2001-2008)
5737                              <1>       ; 'vgabios.c', 'write_gfx_char_pl4'
5738                              <1>
5739                              <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight)
5740                              <1>       ; INPUT ->
5741                              <1>       ; AL = car, BL = attr, DL = xcurs, DH = ycurs,
5742                              <1>       ; [CRT_COLS] = nbcols, [CHAR_HEIGHT] = cheight
5743                              <1>       ; OUTPUT ->
5744                              <1>       ; Regen buffer updated
5745                              <1>
5746 00002E52 51                  <1>       push   ecx
5747 00002E53 53                  <1>       push   ebx
5748 00002E54 52                  <1>       push   edx
5749 00002E55 50                  <1>       push   eax
5750                              <1> wgfxpl_f0:
5751                              <1>       ; switch(cheight)
5752 00002E56 8A25[DE6F0000]      <1>       mov    ah, [CHAR_HEIGHT]
5753 00002E5C 80FC10              <1>       cmp    ah, 16 ; case 16:
5754 00002E5F 7507                <1>       jne    short wgfxpl_f1
5755                              <1>       ; fdata = &vgafont16;
5756 00002E61 BE[A8760100]        <1>       mov    esi, vgafont16
5757 00002E66 EB13                <1>       jmp    short wgfxpl_f3
5758                              <1> wgfxpl_f1:
5759 00002E68 80FC0E              <1>       cmp    ah, 14 ; case 14:
5760 00002E6B 7507                <1>       jne    short wgfxpl_f2
5761 00002E6D BE[A8680100]        <1>       mov    esi, vgafont14
5762 00002E72 EB07                <1>       jmp    short wgfxpl_f3
5763                              <1> wgfxpl_f2:
5764                              <1>       ; default:
5765                              <1>       ;  fdata = &vgafont8;
5766 00002E74 BE[A8600100]        <1>       mov    esi, vgafont8
5767 00002E79 B408                <1>       mov    ah, 8
5768                              <1> wgfxpl_f3:
5769                              <1>       ; al = car
5770 00002E7B F6E4                <1>       mul    ah ; ah = cheight
5771 00002E7D 25FFFF0000          <1>       and    eax, 0FFFFh ; car * cheight
5772                              <1>       ; src = car * cheight;
5773 00002E82 01C6                <1>       add    esi, eax ; esi = fdata[src+i]
5774                              <1>       ; addr=xcurs*8+ycurs*nbcols*64;
5775 00002E84 88F0                <1>       mov    al, dh ; ycurs
5776 00002E86 8A25[DC6F0000]      <1>       mov    ah, [CRT_COLS] ; nbcols
5777 00002E8C F6E4                <1>       mul    ah
5778                              <1>       ; 08/08/2016
5779                              <1>       ;shl   ax, 6 ; * 64
5780 00002E8E 66C1E003            <1>       shl    ax, 3 ; * 8
5781                              <1>       ;sub   dh, dh ; 0
5782                              <1>       ;shl   dx, 3 ; xcurs * 8
5783                              <1>       ;movzx edi, dx
5784 00002E92 0FB6FA              <1>       movzx  edi, dl
5785 00002E95 66C1E703            <1>       shl    di, 3 ; xcurs * 8
5786 00002E99 30F6                <1>       xor    dh, dh
5787 00002E9B 8A15[DE6F0000]      <1>       mov    dl, [CHAR_HEIGHT]
5788 00002EA1 66F7E2              <1>       mul    dx
5789                              <1>       ; eax = ycurs*nbcols*8*[CHAR_HEIGHT]
5790 00002EA4 01C7                <1>       add    edi, eax ; addr
5791 00002EA6 81C700000A00        <1>       add    edi, 0A0000h
5792                              <1>       ;
5793                              <1>       ; outw(VGAREG_SEQU_ADDRESS, 0x0f02);
5794                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
5795 00002EAC 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
5796 00002EB0 66B8020F            <1>       mov    ax, 0F02h
5797 00002EB4 66EF                <1>       out    dx, ax
5798 00002EB6 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
5799 00002EBA 66B80502            <1>       mov    ax, 0205h
5800 00002EBE 66EF                <1>       out    dx, ax
5801                              <1>       ;
5802 00002EC0 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
5803 00002EC4 F6C380              <1>       test   bl, 80h ; if(attr&0x80)
5804 00002EC7 7406                <1>       jz     short wgfxpl_f4 ; else
5805                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x1803);
5806 00002EC9 66B80318            <1>       mov    ax, 1803h
5807 00002ECD EB04                <1>       jmp    short wgfxpl_f5
5808                              <1> wgfxpl_f4:
5809                              <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0003);
5810 00002ECF 66B80300            <1>       mov    ax, 0003h
5811                              <1> wgfxpl_f5:
5812 00002ED3 66EF                <1>       out    dx, ax
5813                              <1>       ;
5814 00002ED5 28FF                <1>       sub    bh, bh ; i = 0
5815                              <1> wgfxpl_0:
5816                              <1>       ; for(i=0;i<cheight;i++)
5817 00002ED7 57                  <1>       push   edi ; addr
5818 00002ED8 0FB605[DC6F0000]    <1>       movzx  eax, byte [CRT_COLS] ; nbcols
5819 00002EDF F6E7                <1>       mul    bh ; nbcols*i
5820                              <1>       ; dest=addr+i*nbcols
5821 00002EE1 01C7                <1>       add    edi, eax ; dest
5822 00002EE3 B580                <1>       mov    ch, 80h ; mask = 0x80;
5823                              <1>       ; for(j=0;j<8;j++)
5824 00002EE5 28C9                <1>       sub    cl, cl ; j = 0
5825                              <1> wgfxpl_1:
5826 00002EE7 D2ED                <1>       shr    ch, cl ; mask=0x80>>j;
5827                              <1>       ;
5828                              <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
```

```
                                <1>       ; read_byte(0xa000,dest);
                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002EE9 88EC                   <1>       mov    ah, ch
00002EEB B008                   <1>       mov    al, 8
00002EED 66EF                   <1>       out    dx, ax
00002EEF 8A07                   <1>       mov    al, [edi] ; ? (io delay?)
                                <1>       ;
00002EF1 28C0                   <1>       sub    al, al ; attr = 0
                                <1>       ; if (fdata[src+i] & mask)
00002EF3 842E                   <1>       test   byte [esi], ch
00002EF5 7404                   <1>       jz     short wgfxpl_2  ; zf = 1
                                <1>       ; write_byte(0xa000,dest,attr&0x0f);
00002EF7 88D8                   <1>       mov    al, bl ; attr;
00002EF9 240F                   <1>       and    al, 0Fh      ; attr&0x0f
                                <1> wgfxpl_2:
                                <1>       ; write_byte(0xa000,dest,0x00);
00002EFB 8807                   <1>       mov    [edi], al ; dest (+ 0A0000h)
00002EFD FEC1                   <1>       inc    cl ; j++
00002EFF 80F908                 <1>       cmp    cl, 8
00002F02 72E3                   <1>       jb     short wgfxpl_1
00002F04 5F                     <1>       pop    edi
                                <1>       ; 08/08/2016
                                <1>       ;cmp   bh, 7
                                <1>       ;jnb   short wgfxpl_3
00002F05 FEC7                   <1>       inc    bh ; i++
00002F07 3A3D[DE6F0000]         <1>       cmp    bh, [CHAR_HEIGHT]
00002F0D 7303                   <1>       jnb    short wgfxpl_3
00002F0F 46                     <1>       inc    esi
00002F10 EBC5                   <1>       jmp    short wgfxpl_0
                                <1> wgfxpl_3:
                                <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00002F12 66B808FF               <1>       mov    ax, 0FF08h
00002F16 66EF                   <1>       out    dx, ax
00002F18 66B80500               <1>       mov    ax, 0005h
00002F1C 66EF                   <1>       out    dx, ax
00002F1E 66B80300               <1>       mov    ax, 0003h
00002F22 66EF                   <1>       out    dx, ax
                                <1>       ;
00002F24 58                     <1>       pop    eax
00002F25 5A                     <1>       pop    edx
00002F26 5B                     <1>       pop    ebx
00002F27 59                     <1>       pop    ecx
00002F28 C3                     <1>       retn
                                <1>
                                <1> vga_write_pixel:
                                <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_write_pixel'
                                <1>
                                <1>       ; INPUT ->
                                <1>       ;      DX = row (0-239)
                                <1>       ;      CX = column (0-799)
                                <1>       ;      AL = pixel value
                                <1>       ;      (AH = [CRT_MODE])
                                <1>       ; OUTPUT ->
                                <1>       ;      none
                                <1>
00002F29 88C3                   <1>       mov    bl, al ; pixel value
                                <1>       ;mov   ah, [CRT_MODE]
00002F2B BE[F66F0000]           <1>       mov    esi, vga_modes
00002F30 89F7                   <1>       mov    edi, esi
00002F32 83C710                 <1>       add    edi, vga_mode_count
                                <1> vga_wp_0:
00002F35 AC                     <1>       lodsb
00002F36 38E0                   <1>       cmp    al, ah ; [CRT_MODE]
00002F38 7405                   <1>       je     short vga_wp_1
00002F3A 39FE                   <1>       cmp    esi, edi
00002F3C 72F7                   <1>       jb     short vga_wp_0
00002F3E C3                     <1>       retn   ; nothing to do
                                <1> vga_wp_1:
00002F3F 83C64F                 <1>       add    esi, vga_memmodel - (vga_modes + 1)
                                <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
00002F42 BF00000A00             <1>       mov    edi, 0A0000h
                                <1>       ;
00002F47 803E04                 <1>       cmp    byte [esi], PLANAR4
00002F4A 741D                   <1>       je     short vga_wp_planar
00002F4C 803E03                 <1>       cmp    byte [esi], PLANAR1
00002F4F 7418                   <1>       je     short vga_wp_planar
                                <1> vga_wp_linear8:
                                <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
00002F51 0FB605[DC6F0000]       <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
00002F58 66C1E003               <1>       shl    ax, 3 ; * 8
00002F5C 66F7E2                 <1>       mul    dx
00002F5F 50                     <1>       push   eax
                                <1>       ;mov   edi, 0A0000h
00002F60 6601CF                 <1>       add    di, cx
00002F63 58                     <1>       pop    eax
00002F64 01C7                   <1>       add    edi, eax ; addr
                                <1>       ; write_byte(0xa000,addr,AL);
00002F66 881F                   <1>       mov    [edi], bl
00002F68 C3                     <1>       retn
                                <1> vga_wp_planar:
                                <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
00002F69 0FB7C1                 <1>       movzx  eax, cx
00002F6C 66C1E803               <1>       shr    ax, 3 ; CX/8
00002F70 50                     <1>       push   eax
00002F71 28E4                   <1>       sub    ah, ah ; 0
00002F73 A0[DC6F0000]           <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
00002F78 66F7E2                 <1>       mul    dx
                                <1>       ;mov   edi, 0A0000h
00002F7B 6601C7                 <1>         add     di, ax
00002F7E 58                     <1>       pop    eax
```

```
5934 00002F7F 01C7              <1>       add    edi, eax ; addr
5935 00002F81 80E107            <1>       and    cl, 7
5936 00002F84 B580              <1>       mov    ch, 80h ; mask
5937 00002F86 D2ED              <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
5938                            <1>
5939                            <1>       ; outw(VGAREG_GRDC_ADDRESS, (mask << 8) | 0x08);
5940 00002F88 66BACE03          <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
5941 00002F8C 88EC              <1>       mov    ah, ch
5942 00002F8E B008              <1>       mov    al, 8
5943 00002F90 66EF              <1>       out    dx, ax
5944                            <1>       ; outw(VGAREG_GRDC_ADDRESS, 0x0205);
5945 00002F92 66B80502          <1>       mov    ax, 0205h
5946 00002F96 66EF              <1>       out    dx, ax
5947                            <1>       ; data = read_byte(0xa000,addr);
5948 00002F98 8A07              <1>       mov    al, [edi] ; (delay?)
5949                            <1>       ; if (AL & 0x80)
5950                            <1>       ; {
5951                            <1>       ;  outw(VGAREG_GRDC_ADDRESS, 0x1803);
5952                            <1>       ; }
5953 00002F9A F6C380            <1>       test   bl, 80h
5954 00002F9D 7406              <1>       jz     short vga_wp_2
5955 00002F9F 66B80318          <1>       mov    ax, 1803h
5956 00002FA3 66EF              <1>       out    dx, ax
5957                            <1> vga_wp_2:
5958                            <1>       ; write_byte(0xa000,addr,AL);
5959 00002FA5 881F              <1>       mov    [edi], bl
5960                            <1>       ;
5961                            <1>       ;mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
5962 00002FA7 66B808FF          <1>       mov    ax, 0FF08h
5963 00002FAB 66EF              <1>       out    dx, ax
5964 00002FAD 66B80500          <1>       mov    ax, 0005h
5965 00002FB1 66EF              <1>       out    dx, ax
5966 00002FB3 66B80300          <1>       mov    ax, 0003h
5967 00002FB7 66EF              <1>       out    dx, ax
5968                            <1>       ;
5969 00002FB9 C3                <1>       retn
5970                            <1>
5971                            <1> vga_read_pixel:
5972                            <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
5973                            <1>       ;
5974                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
5975                            <1>       ; vgabios-0.7a (2011)
5976                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
5977                            <1>       ; 'vgabios.c', 'biosfn_read_pixel'
5978                            <1>
5979                            <1>       ; INPUT ->
5980                            <1>       ;      DX = row (0-239)
5981                            <1>       ;      CX = column (0-799)
5982                            <1>       ;      (AH = [CRT_MODE])
5983                            <1>       ; OUTPUT ->
5984                            <1>       ;      AL = pixel value
5985                            <1>
5986                            <1>       ;mov   ah, [CRT_MODE]
5987 00002FBA BE[F66F0000]      <1>       mov    esi, vga_modes
5988 00002FBF 89F7              <1>       mov    edi, esi
5989 00002FC1 83C710            <1>       add    edi, vga_mode_count
5990                            <1> vga_rp_0:
5991 00002FC4 AC                <1>       lodsb
5992 00002FC5 38E0              <1>       cmp    al, ah ; [CRT_MODE]
5993 00002FC7 7405              <1>       je     short vga_rp_1
5994 00002FC9 39FE              <1>       cmp    esi, edi
5995 00002FCB 72F7              <1>       jb     short vga_rp_0
5996 00002FCD C3                <1>       retn   ; nothing to do
5997                            <1> vga_rp_1:
5998 00002FCE 83C64F            <1>       add    esi, vga_memmodel - (vga_modes + 1)
5999                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6000 00002FD1 BF00000A00        <1>       mov    edi, 0A0000h
6001                            <1>       ;
6002 00002FD6 803E04            <1>       cmp    byte [esi], PLANAR4
6003 00002FD9 741D              <1>       je     short vga_rp_planar
6004 00002FDB 803E03            <1>       cmp    byte [esi], PLANAR1
6005 00002FDE 7418              <1>       je     short vga_rp_planar
6006                            <1> vga_rp_linear8:
6007                            <1>       ; addr=CX+DX*(read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)*8);
6008 00002FE0 0FB605[DC6F0000]  <1>       movzx  eax, byte [CRT_COLS] ; = [VGA_COLS] ; nbcols
6009 00002FE7 66C1E003          <1>       shl    ax, 3 ; * 8
6010 00002FEB 66F7E2            <1>       mul    dx
6011 00002FEE 50                <1>       push   eax
6012                            <1>       ;mov   edi, 0A0000h
6013 00002FEF 6601CF            <1>       add    di, cx
6014 00002FF2 58                <1>       pop    eax
6015 00002FF3 01C7              <1>       add    edi, eax ; addr
6016                            <1>       ; attr=read_byte(0xa000,addr);
6017 00002FF5 8A07              <1>       mov    al, [edi] ; pixel value
6018 00002FF7 C3                <1>       retn
6019                            <1> vga_rp_planar:
6020                            <1>       ; addr = CX/8+DX*read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS);
6021 00002FF8 0FB7C1            <1>       movzx  eax, cx
6022 00002FFB 66C1E803          <1>       shr    ax, 3 ; CX/8
6023 00002FFF 50                <1>       push   eax
6024 00003000 28E4              <1>       sub    ah, ah ; 0
6025 00003002 A0[DC6F0000]      <1>       mov    al, [CRT_COLS] ; = [VGA_COLS] ; nbcols
6026 00003007 66F7E2            <1>       mul    dx
6027                            <1>       ;mov   edi, 0A0000h
6028 0000300A 6601C7            <1>         add     di, ax
6029 0000300D 58                <1>       pop    eax
6030 0000300E 01C7              <1>       add    edi, eax ; addr
6031 00003010 80E107            <1>       and    cl, 7
6032 00003013 B580              <1>       mov    ch, 80h ; mask
6033 00003015 D2ED              <1>       shr    ch, cl       ; mask = 0x80 >> (CX & 0x07);
6034                            <1>       ; attr = 0x00;
6035 00003017 30DB              <1>       xor    bl, bl ; attr = bl = 0,
6036 00003019 30C9              <1>       xor    cl, cl ; i = cl = 0
6037                            <1>       ; for(i=0;i<4;i++)
6038                            <1>             ; {
```

```
                                <1>             ;  outw(VGAREG_GRDC_ADDRESS, (i << 8) | 0x04);
                                <1>             ;  data = read_byte(0xa000,addr) & mask;
                                <1>             ;  if (data > 0) attr |= (0x01 << i);
                                <1>             ; }
                                <1> vga_rp_2:
0000301B 88CC                   <1>       mov   ah, cl ; i << 8
0000301D B004                   <1>       mov   al, 4  ; | 0x04
0000301F 66BACE03               <1>       mov   dx, 3CEh ; VGAREG_GRDC_ADDRESS
00003023 66EF                   <1>       out   dx, ax
                                <1>       ; data = read_byte(0xa000,addr) & mask;
00003025 8A07                   <1>       mov   al, [edi]
00003027 20E8                   <1>       and   al, ch ; & mask
                                <1>       ; if (data > 0) attr |= (0x01 << i);
00003029 08C0                   <1>       or    al, al
0000302B 7408                   <1>       jz    short vga_rp_3 ; al = 0
0000302D B701                   <1>       mov   bh, 1
0000302F D2E7                   <1>       shl   bh, cl ; (0x01 << i)
00003031 08FB                   <1>       or    bl, bh ; attr |= (0x01 << i)
00003033 88D8                   <1>       mov   al, bl ; pixel value
                                <1> vga_rp_3:
00003035 C3                     <1>       retn
                                <1>
                                <1> vga_beeper:
                                <1>       ; 04/08/2016  (TRDOS 386 = TRDOS v2.0)
00003036 FB                     <1>       sti
                                <1>       ;mov  bh, [ACTIVE_PAGE]
00003037 E9BEF3FFFF             <1>         jmp     beeper_gfx
                                <1>
                                <1> vga_write_teletype:
                                <1>       ; 09/12/2017
                                <1>       ; 06/08/2016
                                <1>       ; 04/08/2016
                                <1>       ; 01/08/2016
                                <1>       ; 31/07/2016
                                <1>       ; 09/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_write_teletype'
                                <1>       ; 'biosfn_write_char_only'
                                <1>
                                <1>       ; INPUT ->
                                <1>       ; [CRT_MODE] = current video mode (>7)
                                <1>       ; AL = Character to write
                                <1>       ; BL = Color of character
                                <1>       ; OUTPUT ->
                                <1>       ; Regen buffer updated
                                <1>
                                <1>       ; biosfn_write_teletype (car, page, attr, flag)
                                <1>       ; car = character (AL)
                                <1>       ; page = 0
                                <1>       ; attr = color (BL)
                                <1>       ; 'flag' not used
                                <1>
0000303C 8A25[DA6F0000]         <1>       mov   ah, [CRT_MODE]
00003042 88C7                   <1>       mov   bh, al ; character
00003044 668B15[368A0100]       <1>       mov   dx, [CURSOR_POSN] ; cursor pos for page 0
                                <1>
0000304B BE[FE6F0000]           <1>       mov   esi, vga_g_modes
00003050 89F7                   <1>       mov   edi, esi
00003052 83C708                 <1>       add   edi, vga_g_mode_count
                                <1> vga_wtty_0:
00003055 AC                     <1>       lodsb
00003056 38E0                   <1>       cmp   al, ah ; [CRT_MODE]
00003058 7405                   <1>       je    short vga_wtty_2
0000305A 39FE                   <1>       cmp   esi, edi
0000305C 72F7                   <1>       jb    short vga_wtty_0
                                <1> vga_wtty_1:
0000305E C3                     <1>       retn  ; nothing to do
                                <1> vga_wtty_2:
0000305F 80FF07                 <1>       cmp   bh, 07h ; bell (beep)
00003062 74D2                   <1>       je    short vga_beeper  ; u11
00003064 80FF08                 <1>       cmp   bh, 08h ; backspace
00003067 7508                   <1>       jne   short vga_wtty_3
                                <1>       ; if(xcurs>0)xcurs--;
00003069 08D2                   <1>       or    dl, dl ; xcurs (column)
0000306B 74F1                   <1>       jz    short vga_wtty_1
0000306D FECA                   <1>       dec   dl ; xcurs--;
0000306F EB59                   <1>         jmp     short vga_wtty_12
                                <1> vga_wtty_3:
00003071 80FF0D                 <1>       cmp   bh, 0Dh ; carriage return (\r)
00003074 7504                   <1>       jne   short vga_wtty_4
                                <1>       ; xcurs=0;
00003076 28D2                   <1>       sub   dl, dl ; 0
00003078 EB50                   <1>         jmp     short vga_wtty_12
                                <1> vga_wtty_4:
0000307A 80FF0A                 <1>       cmp   bh, 0Ah ; new line (\n)
0000307D 7504                   <1>       jne   short vga_wtty_5
                                <1>       ; ycurs++;
0000307F FEC6                   <1>       inc   dh ; next row
00003081 EB62                   <1>         jmp     short vga_wtty_11
                                <1> vga_wtty_5:
00003083 80FF09                 <1>       cmp   bh, 09h ; tab stop
00003086 7527                   <1>       jne   short vga_wtty_8
00003088 88D0                   <1>       mov   al, dl
                                <1>       ;cbw
0000308A 30E4                   <1>       xor   ah, ah ; 09/12/2017
0000308C B108                   <1>       mov   cl, 8
0000308E F6F1                   <1>       div   cl
00003090 28E1                   <1>       sub   cl, ah
                                <1>       ;
00003092 B720                   <1>       mov   bh, 20h ; space
                                <1> vga_wtty_6: ; tab stop loop
00003094 6651                   <1>       push  cx
```

```
6144 00003096 6653              <1>       push   bx
6145 00003098 E812000000        <1>       call   vga_wtty_8
6146 0000309D 665B              <1>       pop    bx  ; bh = character, bl = color
6147 0000309F 6659              <1>       pop    cx
6148 000030A1 FEC9              <1>       dec    cl
6149 000030A3 7409              <1>       jz     short vga_wtty_7
6150 000030A5 668B15[368A0100]  <1>       mov    dx, [CURSOR_POSN] ; new cursor position (pg 0)
6151 000030AC EBE6              <1>       jmp    short vga_wtty_6
6152                            <1> vga_wtty_7:
6153 000030AE C3                <1>       retn
6154                            <1>       ;
6155                            <1> vga_wtty_8:
6156 000030AF 83C64F            <1>       add    esi, vga_g_memmodel - (vga_g_modes + 1)
6157                            <1>       ; [ESI] = VGA memory model number (LINEAR8, PLANAR4, PLANAR1)
6158 000030B2 BF00000A00        <1>       mov    edi, 0A0000h
6159                            <1>       ;
6160 000030B7 88F8              <1>       mov    al, bh ; character
6161                            <1>       ;
6162 000030B9 803E04            <1>       cmp    byte [esi], PLANAR4
6163 000030BC 7414              <1>       je     short vga_wtty_planar
6164 000030BE 803E03            <1>       cmp    byte [esi], PLANAR1
6165 000030C1 740F              <1>       je     short vga_wtty_planar
6166                            <1> vga_wtty_linear8:
6167                            <1>       ; write_gfx_char_lin(car,attr,xcurs,ycurs,nbcols);
6168                            <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
6169                            <1>       ; [CRT_COLS] = nbcols
6170 000030C3 E8E7FCFFFF        <1>       call   write_gfx_char_lin
6171 000030C8 EB0D              <1>       jmp    short vga_wtty_9
6172                            <1>
6173                            <1> vga_wtty_12:
6174                            <1>       ; 09/07/2016
6175                            <1>       ; set cursor position
6176                            <1>       ; NOTE: Hardware cursor position will not be set
6177                            <1>       ;   in any VGA modes (>7)
6178                            <1>       ;   But, cursor position will be saved into
6179                            <1>       ;   [CURSOR_POSN].
6180                            <1>       ;   TRDOS 386 (TRDOS v2.0) uses only one page
6181                            <1>       ;   (page 0) for all graphics modes.
6182                            <1>
6183 000030CA 668915[368A0100]  <1>       mov    [CURSOR_POSN], dx ; save cursor pos for pg 0
6184                            <1>       ; 04/08/2016
6185                            <1>       ;mov  bh, [ACTIVE_PAGE] ; = 0
6186                            <1>       ;call _set_cpos
6187 000030D1 C3                <1>       retn
6188                            <1>
6189                            <1> vga_wtty_planar:
6190                            <1>       ; write_gfx_char_pl4(car,attr,xcurs,ycurs,nbcols,cheight);
6191                            <1>       ; AL = car, BL = attr (color), DL = xcurs, DH = ycurs,
6192                            <1>       ; [CRT_COLS]= nbcols, [CHAR_HEIGHT] = cheight
6193 000030D2 E87BFDFFFF        <1>       call   write_gfx_char_pl4
6194                            <1> vga_wtty_9:
6195 000030D7 FEC2              <1>       inc    dl ; xcurs++;
6196                            <1> vga_wtty_10:
6197                            <1>       ; Do we need to wrap ?
6198                            <1>       ; if(xcurs==nbcols)
6199 000030D9 3A15[DC6F0000]    <1>       cmp    dl, [CRT_COLS] ; [VGA_COLS]
6200 000030DF 7204              <1>       jb     short vga_wtty_11 ; no
6201 000030E1 28D2              <1>       sub    dl, dl ; xcurs=0;
6202 000030E3 FEC6              <1>       inc    dh ;  ycurs++;
6203                            <1> vga_wtty_11:
6204                            <1>       ; Do we need to scroll ?
6205                            <1>       ; if(ycurs==nbrows)
6206 000030E5 3A35[E26F0000]    <1>       cmp    dh, [VGA_ROWS]
6207 000030EB 72DD              <1>       jb     short vga_wtty_12 ; no
6208                            <1>       ;
6209                            <1>       ; biosfn_scroll (nblines,attr,rul,cul,rlr,clr,page,dir)
6210                            <1>       ; al = nblines = 1, bl = attr (color) = 0
6211                            <1>       ; ch = rul, cl = cul, dh = rlr, dl = clr, page = 0
6212                            <1>       ; dir = SCROLL_UP
6213                            <1>
6214 000030ED B001              <1>       mov    al, 1
6215 000030EF 28DB              <1>       sub    bl, bl ; 0 ; blank/black line (attr=0) will be used
6216 000030F1 6629C9            <1>       sub    cx, cx ; 0,0
6217                            <1>
6218                            <1>       ; 06/08/2016
6219 000030F4 8A35[E26F0000]    <1>       mov    dh, [VGA_ROWS]
6220 000030FA FECE              <1>       dec    dh ; nbrows -1
6221                            <1>
6222 000030FC 6652              <1>       push   dx    ; 04/08/2016
6223 000030FE 8A15[DC6F0000]    <1>       mov    dl, [CRT_COLS]
6224 00003104 FECA              <1>       dec    dl ; nbcols -1
6225                            <1>
6226 00003106 8A25[DA6F0000]    <1>       mov    ah, [CRT_MODE]
6227                            <1>
6228                            <1>       ; biosfn_scroll(0x01,0x00,0,0,nbrows-1,nbcols-1,page,SCROLL_UP);
6229 0000310C E8FBF4FFFF        <1>       call   vga_graphics_up
6230                            <1>       ; 04/08/2016
6231 00003111 665A              <1>       pop    dx
6232                            <1>       ;dec  dh ; ycurs-=1
6233 00003113 EBB5              <1>       jmp    short vga_wtty_12
6234                            <1>
6235                            <1> font_setup:
6236                            <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
6237                            <1>       ; 09/07/2016
6238                            <1>       ; character generator (font loading) functions
6239                            <1>       ;
6240                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
6241                            <1>       ; vgabios-0.7a (2011)
6242                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
6243                            <1>       ; 'vgabios.c', 'int10_func'
6244                            <1>
6245                            <1>       ; AX = 1100H ; Load User-Defined Font (EGA/VGA)
6246                            <1>       ;
6247                            <1>         ; BH    height of each character (bytes per character definition)
6248                            <1>         ; (BL   font block to load (EGA: 0-3; VGA: 0-7))
```

```
6249                                    <1>       ; CX    number of characters to redefine (<=256)
6250                                    <1>         ; DX    ASCII code of the first character defined at ES:BP
6251                                    <1>         ; EBP      address of font-definition information
6252                                    <1>       ;      (in user's memory space)
6253                                    <1>
6254                                    <1>       ; case 0x11:
6255                                    <1>       ; switch(GET_AL())
6256                                    <1>             ; {
6257                                    <1>       ; case 0x00:
6258                                    <1>         ; case 0x10:
6259                                    <1>         ; biosfn_load_text_user_pat(GET_AL(),ES,BP,CX,DX,GET_BL(),GET_BH());
6260                                    <1>         ; break;
6261                                    <1>
6262                                    <1>       ; AX = 1110H ; Load and Activate User-Defined Font (EGA/VGA)
6263 00003115 08C0                     <1>       or     al, al ; 0
6264 00003117 7404                     <1>       jz     short font_setup_0
6265 00003119 3C10                     <1>       cmp    al, 10h
6266 0000311B 7511                     <1>       jne    short font_setup_1
6267                                    <1> font_setup_0:
6268 0000311D E8CE000000               <1>       call   transfer_user_fonts
6269 00003122 721C                     <1>       jc     short font_setup_error
6270 00003124 E8B2010000               <1>       call   load_text_user_pat
6271 00003129 E931EAFFFF               <1>         jmp     VIDEO_RETURN
6272                                    <1> font_setup_1:
6273                                    <1>       ; AX = 1101H ; Load ROM 8x14 Character Set (EGA/VGA)
6274                                    <1>       ; case 0x01:
6275                                    <1>         ; case 0x11:
6276                                    <1>         ; biosfn_load_text_8_14_pat(GET_AL(),GET_BL());
6277                                    <1>         ; break;
6278 0000312E 3C01                     <1>       cmp    al, 1
6279 00003130 7404                     <1>       je     short font_setup_2
6280 00003132 3C11                     <1>       cmp    al, 11h
6281 00003134 7511                     <1>       jne    short font_setup_3
6282                                    <1> font_setup_2:
6283                                    <1>       ; AX = 1111H ; Load and Activate ROM 8x14 Character Set (EGA/VGA)
6284                                    <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6285 00003136 E8DC020000               <1>       call   load_text_8_14_pat
6286 0000313B E91FEAFFFF               <1>         jmp     VIDEO_RETURN
6287                                    <1> font_setup_error:
6288 00003140 29C0                     <1>       sub    eax, eax ; 0 -> fonts could not be loaded
6289 00003142 E91DEAFFFF               <1>       jmp    _video_return
6290                                    <1> font_setup_3:
6291                                    <1>       ; AX = 1102H ; Load ROM 8x8 Character Set (EGA/VGA)
6292                                    <1>       ; case 0x02:
6293                                    <1>         ; case 0x12:
6294                                    <1>         ; biosfn_load_text_8_8_pat(GET_AL(),GET_BL());
6295                                    <1>         ; break;
6296 00003147 3C02                     <1>       cmp    al, 2
6297 00003149 7404                     <1>       je     short font_setup_4
6298 0000314B 3C12                     <1>       cmp    al, 12h
6299 0000314D 750A                     <1>       jne    short font_setup_5
6300                                    <1> font_setup_4:
6301                                    <1>       ; AX = 1112H ; Load and Activate ROM 8x8 Character Set (EGA/VGA)
6302                                    <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6303 0000314F E8F3020000               <1>       call   load_text_8_8_pat
6304 00003154 E906EAFFFF               <1>         jmp     VIDEO_RETURN
6305                                    <1> font_setup_5:
6306                                    <1>       ; AX = 1104H ; Load ROM 8x16 Character Set (EGA/VGA)
6307                                    <1>       ; case 0x04:
6308                                    <1>         ; case 0x14:
6309                                    <1>         ; biosfn_load_text_8_16_pat(GET_AL(),GET_BL());
6310                                    <1>         ; break;
6311 00003159 3C04                     <1>       cmp    al, 4
6312 0000315B 7404                     <1>       je     short font_setup_6
6313 0000315D 3C14                     <1>       cmp    al, 14h
6314 0000315F 750A                     <1>       jne    short font_setup_7
6315                                    <1> font_setup_6:
6316                                    <1>       ; AX = 1114H ; Load and Activate ROM 8x16 Character Set (EGA/VGA)
6317                                    <1>       ; (BL = font block to load (EGA: 0-3; VGA: 0-7))
6318 00003161 E82A030000               <1>       call   load_text_8_16_pat
6319 00003166 E9F4E9FFFF               <1>         jmp     VIDEO_RETURN
6320                                    <1> font_setup_7:
6321                                    <1>       ; Note: AX=1120h (Setup INT 1Fh, EXT_PTR) is not needed
6322                                    <1>       ; for TRDOS 386 (TRDOS v2.0) video functionality;
6323                                    <1>       ; because, originally EXT_PTR (font address) was used for
6324                                    <1>       ; chars 80h to 0FFh (after the first 128 ASCII char fonts), for
6325                                    <1>       ; CGA graphics mode; currenty, 'vgafont8' address has 256 chars!
6326                                    <1>       ;
6327                                    <1>       ; case 0x20:
6328                                    <1>         ; biosfn_load_gfx_8_8_chars(ES,BP);
6329                                    <1>         ; break;
6330                                    <1>       ; case 0x21:
6331                                    <1>         ; biosfn_load_gfx_user_chars(ES,BP,CX,GET_BL(),GET_DL());
6332                                    <1>         ; break;
6333                                    <1>       ; AX = 1121H ; Setup User-Defined Font for Graphics Mode (VGA)
6334                                    <1>       ; BL   screen rows code: 00H = user-specified (in DL)
6335                                    <1>         ;                      01H = 14 rows
6336                                    <1>         ;                      02H = 25 rows
6337                                    <1>         ;                      03H = 43 rows
6338                                    <1>         ; CX   bytes per character definition
6339                                    <1>         ; DL   (when BL=0) custom number of character rows on screen
6340                                    <1>         ; EBP  address of font-definition information (user's mem space)
6341                                    <1>
6342 0000316B 3C21                     <1>       cmp    al, 21h
6343 0000316D 7531                     <1>       jne    short font_setup_9
6344                                    <1>
6345                                    <1>       ; TRDOS 386 modification !
6346                                    <1>       ; dh = 0 -> 256 characters
6347                                    <1>       ; dh = 80h -> second 128 characters
6348                                    <1>       ; dh = 0FFh -> first 128 characters
6349                                    <1>
6350                                    <1>       ; 09/01/2021 (TRDOS 386 v2.0.3)
6351                                    <1>       ;push  ebx
6352 0000316F 51                       <1>       push   ecx
6353 00003170 52                       <1>       push   edx
```

```
6354 00003171 30D2              <1>       xor    dl, dl
6355 00003173 88CF              <1>       mov    bh, cl ; character height
6356 00003175 66B90001          <1>       mov    cx, 100h ; 256
6357 00003179 08F6              <1>       or     dh, dh ; 0
6358 0000317B 7410              <1>       jz     short  font_setup_8
6359 0000317D FECD              <1>       dec    ch ; cx = 0
6360 0000317F 80FEFF            <1>       cmp    dh, 0FFh
6361 00003182 7409              <1>       je     short font_setup_8 ; 1st 128 chars
6362                            <1>       ; 2nd 128 chars
6363 00003184 80FE80            <1>       cmp    dh, 80h
6364 00003187 75B7              <1>       jne    short font_setup_error ; invalid !
6365 00003189 88F1              <1>       mov    cl, dh
6366 0000318B 86D6              <1>       xchg   dl, dh
6367                            <1>       ; number of chars, cx = 80h
6368                            <1>       ; start char, dl = 80h
6369                            <1> font_setup_8:
6370 0000318D E85E000000        <1>       call   transfer_user_fonts
6371 00003192 5A                <1>       pop    edx
6372 00003193 59                <1>       pop    ecx
6373                            <1>       ;pop   ebx
6374 00003194 72AA              <1>       jc     short font_setup_error
6375                            <1>       ; ebp = user's font data address in system's memory space
6376 00003196 E83F030000        <1>       call   load_gfx_user_chars
6377 0000319B E9BFE9FFFF        <1>         jmp     VIDEO_RETURN
6378                            <1> font_setup_9:
6379                            <1>       ; case 0x22:
6380                            <1>         ; biosfn_load_gfx_8_14_chars(GET_BL());
6381                            <1>         ; break;
6382 000031A0 3C22              <1>       cmp    al, 22h
6383 000031A2 750A              <1>       jne    short font_setup_10
6384 000031A4 E86D030000        <1>       call   load_gfx_8_14_chars
6385 000031A9 E9B1E9FFFF        <1>         jmp     VIDEO_RETURN
6386                            <1> font_setup_10:
6387                            <1>       ; case 0x23:
6388                            <1>         ; biosfn_load_gfx_8_8_dd_chars(GET_BL());
6389                            <1>         ; break;
6390 000031AE 3C23              <1>       cmp    al, 23h
6391 000031B0 750A              <1>       jne    short font_setup_11
6392 000031B2 E8A0030000        <1>       call   load_gfx_8_8_chars
6393 000031B7 E9A3E9FFFF        <1>         jmp     VIDEO_RETURN
6394                            <1> font_setup_11:
6395                            <1>       ; case 0x24:
6396                            <1>         ; biosfn_load_gfx_8_16_chars(GET_BL());
6397                            <1>         ; break;
6398 000031BC 3C24              <1>       cmp    al, 24h
6399 000031BE 750A              <1>       jne    short font_setup_12
6400 000031C0 E8D3030000        <1>       call   load_gfx_8_16_chars
6401 000031C5 E995E9FFFF        <1>         jmp     VIDEO_RETURN
6402                            <1> font_setup_12:
6403                            <1>       ; case 0x30:
6404                            <1>         ; biosfn_get_font_info(GET_BH(),&ES,&BP,&CX,&DX);
6405                            <1>         ; break;
6406 000031CA 3C30              <1>       cmp    al, 30h
6407 000031CC 750A              <1>       jne    short font_setup_13
6408 000031CE E806040000        <1>       call   get_font_info
6409                            <1>       ; eax = return value (info: 4 bytes for 4 parms)
6410                            <1>       ; eax = 0 -> invalid function (input)
6411 000031D3 E98CE9FFFF        <1>         jmp     _video_return
6412                            <1> font_setup_13:
6413 000031D8 3C03              <1>       cmp    al, 03h ; AX = 1103h
6414 000031DA 750D              <1>       jne    short font_setup_14
6415                            <1>       ; biosfn_set_text_block_specifier:
6416                            <1>       ; BL = font block selector code
6417                            <1>       ; NOTE: TRDOS 386 only uses and sets font block 0
6418                            <1>       ; (It is as BL = 0 for TRDOS 386)
6419 000031DC 66BAC403          <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
6420                            <1>       ;mov   ah, bl
6421 000031E0 28E4              <1>       sub    ah, ah ; 0
6422                            <1>       ;mov   al, 03h
6423 000031E2 66EF              <1>       out    dx, ax
6424 000031E4 E976E9FFFF        <1>       jmp     VIDEO_RETURN
6425                            <1>
6426                            <1> font_setup_14:
6427 000031E9 29C0              <1>       sub    eax, eax ; 0 = invalid function
6428 000031EB E974E9FFFF        <1>         jmp     _video_return
6429                            <1>
6430                            <1> transfer_user_fonts:
6431                            <1>       ; 19/01/2021
6432                            <1>       ; 09/01/2021
6433                            <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
6434                            <1>
6435                            <1>       ; BH    height of each character (bytes per character)
6436                            <1>       ; CX    number of characters to redefine (<=256)
6437                            <1>         ; DX    ASCII code of the first character defined at EBP
6438                            <1>         ; EBP      address of font-definition information
6439                            <1>       ;      (in user's memory space)
6440                            <1>
6441                            <1>       ; Modified registers: eax, edx, ecx, esi, edi, ebp
6442                            <1>       ;
6443                            <1>       ; output:
6444                            <1>       ;      ebp = user font data address in system memory
6445                            <1>
6446 000031F0 81E2FFFF0000      <1>       and    edx, 0FFFFh
6447 000031F6 81E1FFFF0000      <1>       and    ecx, 0FFFFh
6448 000031FC 7537              <1>       jnz    short transfer_user_fonts_5
6449                            <1>
6450 000031FE 09D2              <1>       or     edx, edx
6451 00003200 7531              <1>       jnz    short transfer_user_fonts_4
6452 00003202 09ED              <1>       or     ebp, ebp
6453 00003204 752D              <1>       jnz    short transfer_user_fonts_4
6454                            <1>
6455                            <1>       ; cx = 0, dx = 0, ebp = 0
6456                            <1>       ; copy system font to user font
6457                            <1>
6458 00003206 B140              <1>       mov    cl, 64 ; 64 dwords
```

```
6459                                    <1>
6460 00003208 80FF10                    <1>       cmp    bh, 16
6461 0000320B 7417                      <1>       je     short transfer_user_fonts_2
6462 0000320D 80FF08                    <1>       cmp    bh, 8
6463 00003210 7406                      <1>       je     short transfer_user_fonts_1
6464                                    <1>
6465 00003212 BD[A8680100]              <1>       mov    ebp, vgafont14
6466 00003217 C3                        <1>       retn
6467                                    <1>
6468                                    <1> transfer_user_fonts_1:
6469 00003218 BF00500900                <1>       mov    edi, VGAFONT8USER
6470 0000321D BE[A8600100]              <1>       mov    esi, vgafont8
6471 00003222 EB0A                      <1>       jmp    short transfer_user_fonts_3
6472                                    <1>
6473                                    <1> transfer_user_fonts_2:
6474 00003224 BF00400900                <1>       mov    edi, VGAFONT16USER
6475 00003229 BE[A8760100]              <1>       mov    esi, vgafont16
6476                                    <1> transfer_user_fonts_3:
6477 0000322E 89FD                      <1>       mov    ebp, edi
6478 00003230 F3A5                      <1>       rep    movsd
6479 00003232 C3                        <1>       retn
6480                                    <1>
6481                                    <1> transfer_user_fonts_4:
6482 00003233 F9                        <1>       stc
6483 00003234 C3                        <1>       retn
6484                                    <1>
6485                                    <1> transfer_user_fonts_5:
6486 00003235 09ED                      <1>       or     ebp, ebp
6487 00003237 74FA                      <1>       jz     short transfer_user_fonts_4 ; invalid address !
6488                                    <1>
6489 00003239 6681F90001                <1>       cmp    cx, 256
6490 0000323E 77F3                      <1>       ja     short transfer_user_fonts_4
6491 00003240 29D1                      <1>       sub    ecx, edx
6492 00003242 76EF                      <1>       jna    short transfer_user_fonts_4
6493                                    <1>
6494 00003244 80FF0E                    <1>       cmp    bh, 14 ; 8x14 font
6495                                    <1>                     ; (there is not an alternative buffer)
6496 00003247 7525                      <1>       jne    short transfer_user_fonts_6
6497                                    <1>
6498                                    <1>       ; use system's 8x14 font space if permission flag is 1
6499 00003249 F605[7E120300]80          <1>       test   byte [ufont], 80h
6500 00003250 74E1                      <1>       jz     short transfer_user_fonts_4 ; not allowed
6501                                    <1>
6502                                    <1>       ; permission is given (for vgafont14 location etc.)
6503                                    <1>       ; (for permanent font modification)
6504                                    <1>       ;
6505                                    <1>       ; 19/01/2021
6506                                    <1>       ; Note: Permission flag can be set by 'root' while
6507                                    <1>       ;     system is not in multi tasking/user mode
6508                                    <1>       ;     while [multi_tasking] = 0 and [u.uid] = 0
6509                                    <1>
6510 00003252 52                        <1>       push   edx
6511 00003253 30E4                      <1>       xor    ah, ah
6512 00003255 88F8                      <1>       mov    al, bh ; mov al, 14
6513 00003257 66F7E1                    <1>       mul    cx
6514                                    <1>              ; ecx = byte count
6515 0000325A 89C1                      <1>       mov    ecx, eax
6516 0000325C 5A                        <1>       pop    edx
6517 0000325D 30E4                      <1>       xor    ah, ah
6518 0000325F 88F8                      <1>       mov    al, bh ; mov ax, 14 ; bytes per character
6519 00003261 66F7E2                    <1>       mul    dx
6520 00003264 6689C2                    <1>       mov    dx, ax ; char offset
6521 00003267 BF[A8680100]              <1>       mov    edi, vgafont14
6522 0000326C EB4C                      <1>       jmp    short transfer_user_fonts_8
6523                                    <1> transfer_user_fonts_6:
6524 0000326E 80FF08                    <1>       cmp    bh, 8 ; 8x8 font
6525 00003271 7522                      <1>       jne    short transfer_user_fonts_7 ; 8x16 font
6526 00003273 66C1E203                  <1>       shl    dx, 3 ; * 8
6527 00003277 66C1E103                  <1>       shl    cx, 3 ; * 8
6528                                    <1>       ; 09/01/2021
6529 0000327B BF00500900                <1>       mov    edi, VGAFONT8USER
6530 00003280 F605[7E120300]08          <1>       test   byte [ufont], 8  ; already loaded ?
6531 00003287 7531                      <1>       jnz    short transfer_user_fonts_8 ; yes
6532 00003289 BE[A8600100]              <1>       mov    esi, vgafont8
6533 0000328E E83B000000                <1>       call   transfer_user_fonts_10
6534 00003293 EB25                      <1>       jmp    short transfer_user_fonts_8
6535                                    <1> transfer_user_fonts_7:
6536 00003295 80FF10                    <1>       cmp    bh, 16 ; 8x16 font
6537 00003298 7599                      <1>       jne    short transfer_user_fonts_4  ; invalid !
6538 0000329A 66C1E204                  <1>       shl    dx, 4 ; * 16
6539 0000329E 66C1E104                  <1>       shl    cx, 4 ; * 16
6540 000032A2 BF00400900                <1>       mov    edi, VGAFONT16USER
6541 000032A7 F605[7E120300]10          <1>       test   byte [ufont], 16  ; already loaded ?
6542 000032AE 750A                      <1>       jnz    short transfer_user_fonts_8 ; yes
6543 000032B0 BE[A8760100]              <1>       mov    esi, vgafont16
6544 000032B5 E814000000                <1>       call   transfer_user_fonts_10
6545                                    <1> transfer_user_fonts_8:
6546 000032BA 01D7                      <1>       add    edi, edx ; char offset
6547                                    <1>       ; 09/07/2016
6548                                    <1>       ;and   ecx, 0FFFFh
6549                                    <1>       ; ECX = byte count
6550                                    <1>       ;push  ecx
6551 000032BC 89EE                      <1>       mov    esi, ebp ; user's font buffer
6552                                    <1>       ; 09/01/2021
6553 000032BE 89FD                      <1>       mov    ebp, edi ; system addr for user's font
6554                                    <1>       ; 05/01/2021
6555                                    <1>       ;mov   edi, Cluster_Buffer  ; system buffer
6556 000032C0 E8CCE80000                <1>       call   transfer_from_user_buffer
6557                                    <1>       ;pop   ecx
6558                                    <1>       ; ecx = transfer (byte) count = character count
6559 000032C5 7206                      <1>       jc     short transfer_user_fonts_9
6560                                    <1>       ; 05/01/2021
6561                                    <1>       ;mov   ebp, Cluster_Buffer
6562                                    <1>
6563 000032C7 083D[7E120300]            <1>       or     byte [ufont], bh
```

```
                                <1>                     ; 8x8 or 8x16 user font ready
                                <1> transfer_user_fonts_9:
000032CD C3                     <1>     retn
                                <1>
                                <1> transfer_user_fonts_10:
                                <1>     ; 09/01/2021
000032CE 56                     <1>     push   esi
000032CF 57                     <1>     push   edi
000032D0 51                     <1>     push   ecx
000032D1 66B94000               <1>     mov    cx, 64
000032D5 F3A5                   <1>     rep    movsd
000032D7 59                     <1>     pop    ecx
000032D8 5F                     <1>     pop    edi
000032D9 5E                     <1>     pop    esi
000032DA C3                     <1>     retn
                                <1>
                                <1> load_text_user_pat:
                                <1>     ; 26/07/2016
                                <1>     ; 09/07/2016
                                <1>     ; load user defined (EGA/VGA) text fonts
                                <1>     ;
                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
                                <1>     ; vgabios-0.7a (2011)
                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
                                <1>     ; 'vgabios.c', 'biosfn_load_text_user_pat'
                                <1>
                                <1>     ; biosfn_load_text_user_pat (AL,ES,BP,CX,DX,BL,BH)
                                <1>
                                <1>     ; get_font_access();
                                <1>     ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>     ; for(i=0;i<CX;i++)
                                <1>     ; {
                                <1>     ;  src = BP + i * BH;
                                <1>     ;  dest = blockaddr + (DX + i) * 32;
                                <1>     ;  memcpyb(0xA000, dest, ES, src, BH);
                                <1>     ; }
                                <1>     ; release_font_access();
                                <1>     ; if(AL>=0x10)
                                <1>     ; {
                                <1>     ; set_scan_lines(BH);
                                <1>     ; }
                                <1>
000032DB 50                     <1>     push   eax
000032DC E83C000000             <1>     call   get_font_access
000032E1 28DB                   <1>     sub    bl, bl ; i = 0
                                <1> ltup_1:
000032E3 88D8                   <1>     mov    al, bl
000032E5 F6E7                   <1>     mul    bh
000032E7 0FB7F0                 <1>     movzx  esi, ax
000032EA 01EE                   <1>     add    esi, ebp
000032EC 88D8                   <1>     mov    al, bl
000032EE 28E4                   <1>     sub    ah, ah
000032F0 6601D0                 <1>     add    ax, dx ; (DX + i)
000032F3 66C1E005               <1>     shl    ax, 5  ; * 32
000032F7 0FB7F8                 <1>     movzx  edi, ax
000032FA 81C700000A00           <1>     add    edi, 0A0000h
00003300 51                     <1>     push   ecx
00003301 0FB6CF                 <1>     movzx  ecx, bh
00003304 F3A4                   <1>     rep    movsb
00003306 59                     <1>     pop    ecx
00003307 FEC3                   <1>     inc    bl
00003309 38CB                   <1>     cmp    bl, cl
0000330B 75D6                   <1>     jne    short ltup_1
                                <1>     ;
0000330D E840000000             <1>     call   release_font_access
                                <1>     ;
00003312 58                     <1>     pop    eax
                                <1>     ; if(AL>=0x10)
00003313 3C10                   <1>     cmp    al, 10h
00003315 7205                   <1>     jb     short ltup_2
                                <1>     ; set_scan_lines(BH);
00003317 E875000000             <1>     call   set_scan_lines
                                <1> ltup_2:
0000331C C3                     <1>     retn
                                <1>
                                <1> get_font_access:
                                <1>     ; 09/07/2016
                                <1>     ;
                                <1>     ; derived from 'Plex86/Bochs VGABios' source code
                                <1>     ; vgabios-0.7a (2011)
                                <1>     ; by the LGPL VGABios developers Team (2001-2008)
                                <1>     ; 'vgabios.c', 'get_font_access'
                                <1>
                                <1>     ; get_font_access()
0000331D 52                     <1>     push   edx
0000331E 66BAC403               <1>     mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00003322 66B80001               <1>     mov    ax, 0100h
00003326 66EF                   <1>     out    dx, ax
00003328 66B80204               <1>     mov    ax, 0402h
0000332C 66EF                   <1>     out    dx, ax
0000332E 66B80407               <1>     mov    ax, 0704h
00003332 66EF                   <1>     out    dx, ax
00003334 66B80003               <1>     mov    ax, 0300h
00003338 66EF                   <1>     out    dx, ax
0000333A 66BACE03               <1>     mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
0000333E 66B80402               <1>     mov    ax, 0204h
00003342 66EF                   <1>     out    dx, ax
00003344 66B80500               <1>     mov    ax, 0005h
00003348 66EF                   <1>     out    dx, ax
0000334A 66B80604               <1>     mov    ax, 0406h
0000334E 66EF                   <1>     out    dx, ax
00003350 5A                     <1>     pop    edx
00003351 C3                     <1>     retn
                                <1>
                                <1> release_font_access:
```

```
                        <1>       ; 29/07/2016
                        <1>       ; 09/07/2016
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'release_font_access'
                        <1>
00003352 66BAC403       <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
00003356 66B80001       <1>       mov    ax, 0100h
0000335A 66EF           <1>       out    dx, ax
0000335C 66B80203       <1>       mov    ax, 0302h
00003360 66EF           <1>       out    dx, ax
00003362 66B80403       <1>       mov    ax, 0304h
00003366 66EF           <1>       out    dx, ax
00003368 66B80003       <1>       mov    ax, 0300h
0000336C 66EF           <1>       out    dx, ax
0000336E 66BACC03       <1>       mov    dx, 3CCh ; VGAREG_READ_MISC_OUTPUT
00003372 EC             <1>       in     al, dx
00003373 2401           <1>       and    al, 01h
00003375 C0E002         <1>       shl    al, 2
00003378 0C0A           <1>       or     al, 0Ah
0000337A 88C4           <1>       mov    ah, al
0000337C B006           <1>       mov    al, 06h
0000337E 66BACE03       <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
00003382 66EF           <1>       out    dx, ax
00003384 66B80400       <1>       mov    ax, 0004h
00003388 66EF           <1>       out    dx, ax
0000338A 66B80510       <1>       mov    ax, 1005h
0000338E 66EF           <1>       out    dx, ax
00003390 C3             <1>       retn
                        <1>
                        <1> set_scan_lines:
                        <1>       ; 09/07/2016
                        <1>       ;
                        <1>       ; derived from 'Plex86/Bochs VGABios' source code
                        <1>       ; vgabios-0.7a (2011)
                        <1>       ; by the LGPL VGABios developers Team (2001-2008)
                        <1>       ; 'vgabios.c', 'set_scan_lines'
                        <1>
                        <1>       ; set_scan_lines(lines)
                        <1>       ; BH = lines
                        <1>
                        <1>       ; outb(crtc_addr, 0x09);
00003391 66BAD403       <1>       mov    dx, 3D4h ; CRTC_ADDRESS = 3D4h (always)
00003395 B009           <1>       mov    al, 09h
00003397 EE             <1>       out    dx, al
                        <1>       ; crtc_r9 = inb(crtc_addr+1);
00003398 6642           <1>       inc    dx ; 3D5h
0000339A EC             <1>       in     al, dx
                        <1>       ; crtc_r9 = (crtc_r9 & 0xe0) | (lines - 1);
0000339B 24E0           <1>       and    al, 0E0h
0000339D FECF           <1>       dec    bh ; lines - 1
0000339F 08F8           <1>       or     al, bh
                        <1>       ; outb(crtc_addr+1, crtc_r9);
000033A1 EE             <1>       out    dx, al
                        <1>       ;inc  bh
                        <1>       ; if(lines==8)
                        <1>       ;cmp  bh, 8
000033A2 80FF07         <1>       cmp    bh, 7
000033A5 7506           <1>       jne    short ssl_1
                        <1>       ; biosfn_set_cursor_shape(0x06,0x07);
000033A7 66B90706       <1>       mov    cx, 0607h
000033AB EB06           <1>       jmp    short ssl_2
                        <1> ssl_1:
                        <1>       ; biosfn_set_cursor_shape(lines-4,lines-3);
000033AD 88F9           <1>       mov    cl, bh ; lines - 1
000033AF 88CD           <1>       mov    ch, cl ; lines - 1 (16 -> 15)
000033B1 FECD           <1>       dec    ch  ; lines - 2 (16 -> 14)
                        <1> ssl_2:
                        <1>       ; CH = start line, CL = stop line
000033B3 B40A           <1>       mov    ah, 10 ; 6845 register for cursor set
000033B5 66890D[F36F0000]   <1>   mov    [CURSOR_MODE], cx ; save in data area
000033BC E809F0FFFF     <1>       call   m16    ; output cx register
                        <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, lines);
000033C1 FEC7           <1>       inc    bh ; lines
000033C3 883D[DE6F0000]     <1>   mov    [CHAR_HEIGHT], bh
                        <1>       ;  outb(crtc_addr, 0x12);
000033C9 66BAD403       <1>       mov    dx, 3D4h ; CRTC_ADDRESS
000033CD B012           <1>       mov    al, 12h
000033CF EE             <1>       out    dx, al
                        <1>       ; vde = inb(crtc_addr+1);
000033D0 6642           <1>       inc    dx
000033D2 EC             <1>       in     al, dx
000033D3 88C4           <1>       mov    ah, al
                        <1>       ; outb(crtc_addr, 0x07);
000033D5 664A           <1>       dec    dx
000033D7 B007           <1>       mov    al, 07h
000033D9 EE             <1>       out    dx, al
                        <1>       ; ovl = inb(crtc_addr+1);
000033DA 6642           <1>       inc    dx
000033DC EC             <1>       in     al, dx
                        <1>       ; vde += (((ovl & 0x02) << 7) + ((ovl & 0x40) << 3) + 1);
000033DD 88E2           <1>       mov    dl, ah ; vde
000033DF 88C6           <1>       mov    dh, al ; ovl
000033E1 6683E002       <1>       and    ax, 02h
000033E5 66C1E007       <1>       shl    ax, 7
000033E9 6689C1         <1>       mov    cx, ax ; (ovl & 0x02) << 7)
000033EC 88F0           <1>       mov    al, dh ; ovl
000033EE 6683E040       <1>       and    ax, 40h
000033F2 66C1E003       <1>       shl    ax, 3  ; (ovl & 0x40) << 3)
000033F6 6640           <1>       inc    ax ; + 1
000033F8 6601C8         <1>       add    ax, cx
000033FB 30F6           <1>       xor    dh, dh
000033FD 6601D0         <1>       add    ax, dx ; + vde
```

```
                                <1>       ; rows = vde / lines;
00003400 F6F7                   <1>       div    bh
                                <1>       ;dec   al ; rows -1
                                <1>       ; write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, rows-1);
00003402 A2[E26F0000]           <1>       mov    [VGA_ROWS], al ; rows (not 'rows-1' !)
                                <1>       ; write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, rows * cols * 2);
                                <1>       ;mov   ah, [CRT_COLS]
                                <1>       ;mul   ah
                                <1>       ; 17/11/2020
00003407 F625[DC6F0000]         <1>       mul    byte [CRT_COLS]
0000340D 66D1E0                 <1>       shl    ax, 1
00003410 66A3[B0960100]         <1>       mov    [CRT_LEN], ax
00003416 C3                     <1>       retn
                                <1>
                                <1> load_text_8_14_pat:
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; 23/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_14_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_14_pat (AL,BL)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 14;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont14+src, 14);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(14);
                                <1>       ; }
                                <1>
00003417 50                     <1>       push   eax
00003418 E800FFFFFF             <1>       call   get_font_access
                                <1>
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ;mov   dl, bl
                                <1>       ;and   dl, 3
                                <1>       ;shl   dx, 14
                                <1>       ;xchg  dx, bx
                                <1>       ;and   dl, 4
                                <1>       ;shl   dx, 11
                                <1>       ;add   dx, bx
                                <1>
                                <1>       ;xor   dx, dx  ; blockaddr = 0
                                <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
                                <1>
0000341D 28DB                   <1>       sub    bl, bl ; i = 0
0000341F B70E                   <1>       mov    bh, 14
00003421 BE[A8680100]           <1>       mov    esi, vgafont14
00003426 BF00000A00             <1>       mov    edi, 0A0000h
                                <1> lt8_14_1:
                                <1>       ;mov   al, bl
                                <1>       ;mul   bh
                                <1>       ;movzx esi, ax
                                <1>       ;add   esi, vgafont14
                                <1>       ;mov   al, bl
                                <1>       ;sub   ah, ah
                                <1>       ;shl   ax, 5 ; * 32
                                <1>       ;;add  ax, dx ; blockaddr + i * 32;
                                <1>       ;movzx edi, ax ; dest
                                <1>       ;add   edi, 0A0000h
0000342B 0FB6CF                 <1>       movzx  ecx, bh
0000342E F3A4                   <1>       rep    movsb
00003430 83C712                 <1>       add    edi, 18 ; 32 - 14
00003433 FEC3                   <1>       inc    bl
00003435 75F4                   <1>       jnz    short lt8_14_1
                                <1>       ;
00003437 E816FFFFFF             <1>       call   release_font_access
                                <1>       ;
0000343C 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
0000343D 3C10                   <1>       cmp    al, 10h
0000343F 7205                   <1>       jb     short lt8_14_4
                                <1>       ; BH = 14
                                <1>       ; set_scan_lines(14);
00003441 E84BFFFFFF             <1>       call   set_scan_lines
                                <1> lt8_14_4:
00003446 C3                     <1>       retn
                                <1>
                                <1> load_text_8_8_pat:
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; 23/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_8_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_8_pat (AL,BL)
                                <1>
```

```
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 8;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont8+src, 8);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(8);
                                <1>       ; }
                                <1>
00003447 50                     <1>       push   eax
00003448 E8D0FEFFFF             <1>       call   get_font_access
                                <1>
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ;mov   dl, bl
                                <1>       ;and   dl, 3
                                <1>       ;shl   dx, 14
                                <1>       ;xchg  dx, bx
                                <1>       ;and   dl, 4
                                <1>       ;shl   dx, 11
                                <1>       ;add   dx, bx
                                <1>
                                <1>       ;xor   dx, dx  ; blockaddr = 0
                                <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
                                <1>
0000344D 28DB                   <1>       sub    bl, bl ; i = 0
0000344F B708                   <1>       mov    bh, 8
                                <1>       ;mov   esi, vgafont8
00003451 BF00000A00             <1>       mov    edi, 0A0000h
                                <1>
                                <1>       ; 05/01/2021
00003456 F605[7E120300]80       <1>       test   byte [ufont], 80h
0000345D 7410                   <1>       jz     short lt8_8_0
                                <1>              ; user font permission (after set mode)
0000345F F605[7E120300]08       <1>       test   byte [ufont], 8
00003466 7407                   <1>       jz     short lt8_8_0
00003468 BE00500900             <1>       mov    esi, VGAFONT8USER
0000346D EB05                   <1>       jmp    short lt8_8_1
                                <1> lt8_8_0:
0000346F BE[A8600100]           <1>       mov    esi, vgafont8
                                <1> lt8_8_1:
                                <1>       ;mov   al, bl
                                <1>       ;mul   bh
                                <1>       ;movzx esi, ax
                                <1>       ;add   esi, vgafont8
                                <1>       ;mov   al, bl
                                <1>       ;sub   ah, ah
                                <1>       ;shl   ax, 5 ; * 32
                                <1>       ;;add  ax, dx ; blockaddr + i * 32;
                                <1>       ;movzx edi, ax ; dest
                                <1>       ;add   edi, 0A0000h
00003474 0FB6CF                 <1>       movzx  ecx, bh
00003477 F3A4                   <1>       rep    movsb
00003479 83C718                 <1>       add    edi, 24 ; 32 - 8
0000347C FEC3                   <1>       inc    bl
0000347E 75F4                   <1>       jnz    short lt8_8_1
                                <1>       ;
00003480 E8CDFEFFFF             <1>       call   release_font_access
                                <1>       ;
00003485 58                     <1>       pop    eax
                                <1>       ; if(AL>=0x10)
00003486 3C10                   <1>       cmp    al, 10h
00003488 7205                   <1>       jb     short lt8_8_2
                                <1>       ; BH = 8
                                <1>       ; set_scan_lines(8);
0000348A E802FFFFFF             <1>       call   set_scan_lines
                                <1> lt8_8_2:
0000348F C3                     <1>       retn
                                <1>
                                <1> load_text_8_16_pat:
                                <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; 26/07/2016
                                <1>       ; 25/07/2016
                                <1>       ; 23/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; load user defined (EGA/VGA) text fonts
                                <1>       ;
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios developers Team (2001-2008)
                                <1>       ; 'vgabios.c', 'biosfn_load_text_8_16_pat'
                                <1>
                                <1>       ; biosfn_load_text_8_16_pat (AL,BL)
                                <1>
                                <1>       ; get_font_access();
                                <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
                                <1>       ; for(i=0;i<0x100;i++)
                                <1>       ; {
                                <1>       ;  src = i * 16;
                                <1>       ;  dest = blockaddr + i * 32;
                                <1>       ;  memcpyb(0xA000, dest, 0xC000, vgafont16+src, 16);
                                <1>       ; }
                                <1>       ; release_font_access();
                                <1>       ; if(AL>=0x10)
                                <1>       ; {
                                <1>       ; set_scan_lines(16);
                                <1>       ; }
                                <1>
00003490 50                     <1>       push   eax
00003491 E887FEFFFF             <1>       call   get_font_access
                                <1>
```

```
6984                                    <1>       ; blockaddr = ((BL & 0x03) << 14) + ((BL & 0x04) << 11);
6985                                    <1>       ;mov   dl, bl
6986                                    <1>       ;and   dl, 3
6987                                    <1>       ;shl   dx, 14
6988                                    <1>       ;xchg  dx, bx
6989                                    <1>       ;and   dl, 4
6990                                    <1>       ;shl   dx, 11
6991                                    <1>       ;add   dx, bx
6992                                    <1>
6993                                    <1>       ;xor   dx, dx  ; blockaddr = 0
6994                                    <1>       ; Always block 0 for TRDOS 386 ! (blockaddr=0)
6995                                    <1>
6996 00003496 28DB                     <1>       sub    bl, bl ; i = 0
6997 00003498 B710                     <1>       mov    bh, 16
6998                                    <1>       ;mov   esi, vgafont16
6999 0000349A BF00000A00               <1>       mov    edi, 0A0000h
7000 0000349F 0FB6C7                   <1>       movzx  eax, bh
7001                                    <1>
7002                                    <1>       ; 05/01/2021
7003 000034A2 F605[7E120300]80         <1>       test   byte [ufont], 80h
7004 000034A9 7410                     <1>       jz     short lt8_16_0
7005                                    <1>              ; user font permission (after set mode)
7006 000034AB F605[7E120300]10         <1>       test   byte [ufont], 16
7007 000034B2 7407                     <1>       jz     short lt8_16_0
7008 000034B4 BE00400900               <1>       mov    esi, VGAFONT16USER
7009 000034B9 EB05                     <1>       jmp    short lt8_16_1
7010                                    <1> lt8_16_0:
7011 000034BB BE[A8760100]             <1>       mov    esi, vgafont16
7012                                    <1> lt8_16_1:
7013                                    <1>       ;mov   al, bl
7014                                    <1>       ;mul   bh
7015                                    <1>       ;movzx esi, ax
7016                                    <1>       ;add   esi, vgafont16
7017                                    <1>       ;mov   al, bl ; i
7018                                    <1>       ;sub   ah, ah
7019                                    <1>       ;shl   ax, 5 ; * 32
7020                                    <1>       ;;add  ax, dx ; blockaddr + i * 32;
7021                                    <1>       ;movzx edi, ax ; dest
7022                                    <1>       ;add   edi, 0A0000h
7023                                    <1>       ;movzx ecx, bh
7024 000034C0 89C1                     <1>       mov    ecx, eax ; 16
7025 000034C2 F3A4                     <1>       rep    movsb
7026 000034C4 01C7                     <1>       add    edi, eax ; add edi, 16
7027 000034C6 FEC3                     <1>       inc    bl
7028 000034C8 75F6                     <1>       jnz    short lt8_16_1
7029                                    <1>       ;
7030 000034CA E883FEFFFF               <1>       call   release_font_access
7031                                    <1>       ;
7032 000034CF 58                       <1>       pop    eax
7033                                    <1>       ; if(AL>=0x10)
7034 000034D0 3C10                     <1>       cmp    al, 10h
7035 000034D2 7205                     <1>       jb     short lt8_16_2
7036                                    <1>       ; BH = 16
7037                                    <1>       ; set_scan_lines(16);
7038 000034D4 E8B8FEFFFF               <1>       call   set_scan_lines
7039                                    <1> lt8_16_2:
7040 000034D9 C3                       <1>       retn
7041                                    <1>
7042                                    <1> load_gfx_user_chars:
7043                                    <1>       ; 08/01/2021
7044                                    <1>       ; 05/01/2021 (TRDOS 386 v2.0.3)
7045                                    <1>       ; 08/08/2016
7046                                    <1>       ; 10/07/2016
7047                                    <1>       ; Setup User-Defined Font for Graphics Mode (VGA)
7048                                    <1>       ;
7049                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7050                                    <1>       ; vgabios-0.7a (2011)
7051                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7052                                    <1>       ; 'vgabios.c', 'biosfn_load_gfx_user_chars'
7053                                    <1>
7054                                    <1>       ; biosfn_load_gfx_user_chars (ES,BP,CX,BL,DL)
7055                                    <1>       ; /* set 0x43 INT pointer */
7056                                    <1>       ; write_word(0x0, 0x43*4, BP);
7057                                    <1>       ; write_word(0x0, 0x43*4+2, ES);
7058                                    <1>
7059                                    <1>       ; 08/01/2021
7060                                    <1>
7061                                    <1>       ; BL   screen rows code: 00H = user-specified (in DL)
7062                                    <1>         ;                      01H = 14 rows
7063                                    <1>         ;                      02H = 25 rows
7064                                    <1>         ;                      03H = 43 rows
7065                                    <1>         ; CX   bytes per character definition
7066                                    <1>         ; DL   (when BL=0) custom number of character rows on screen
7067                                    <1>         ; EBP  address of font-definition information (user's mem space)
7068                                    <1>
7069                                    <1>       ; 05/01/2021
7070                                    <1>       ;xor   eax, eax
7071                                    <1>       ;dec   eax ; 0FFFFFFFFh (user defined fonts)
7072                                    <1>       ;mov   [VGA_INT43H], eax
7073                                    <1>
7074                                    <1>       ; 08/01/2021
7075                                    <1>       ; ebp = video font data (buffer) address
7076 000034DA 892D[C2960100]           <1>       mov    [VGA_INT43H], ebp
7077                                    <1>
7078                                    <1>       ; switch (BL) {
7079                                    <1>       ; case 0:
7080                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7081                                    <1>       ;    break;
7082 000034E0 20DB                     <1>       and    bl, bl
7083 000034E2 7508                     <1>       jnz    short l_gfx_uc_1
7084 000034E4 8815[E26F0000]           <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
7085 000034EA EB23                     <1>       jmp    short l_gfx_uc_4
7086                                    <1> l_gfx_uc_1:
7087                                    <1>       ; case 1:
7088                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
```

```
                                <1>       ;    break;
000034EC FECB                   <1>       dec   bl
000034EE 7509                   <1>       jnz   short l_gfx_uc_2
                                <1>       ; bl = 1
000034F0 C605[E26F0000]0E       <1>       mov   byte [VGA_ROWS], 14  ; not 13 !
000034F7 EB16                   <1>       jmp   short l_gfx_uc_4
                                <1> l_gfx_uc_2:
000034F9 FECB                   <1>       dec   bl
000034FB 740B                   <1>       jz    short l_gfx_uc_3 ; bl = 2
000034FD FECB                   <1>       dec   bl
000034FF 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;
00003501 C605[E26F0000]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
00003508 C605[E26F0000]19       <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
                                <1>       ; }
                                <1> l_gfx_uc_4:
                                <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, CX);
0000350F 880D[DE6F0000]         <1>       mov   [CHAR_HEIGHT], cl
                                <1>       ; }
00003515 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);
00003516 C705[C2960100]-        <1>       mov   dword [VGA_INT43H], vgafont14
0000351C [A8680100]             <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;
00003520 20DB                   <1>       and   bl, bl
00003522 7508                   <1>       jnz   short l_gfx_8_14c_1
00003524 8815[E26F0000]         <1>       mov   [VGA_ROWS], dl  ; not DL-1 !
0000352A 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;
0000352C FECB                   <1>       dec   bl
0000352E 7509                   <1>       jnz   short l_gfx_8_14c_2
                                <1>       ; bl = 1
00003530 C605[E26F0000]0E       <1>       mov   byte [VGA_ROWS], 14  ; not 13 !
00003537 EB16                   <1>       jmp   short l_gfx_8_14c_4
                                <1> l_gfx_8_14c_2:
00003539 FECB                   <1>       dec   bl
0000353B 740B                   <1>       jz    short l_gfx_8_14c_3 ; bl = 2
0000353D FECB                   <1>       dec   bl
0000353F 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;
00003541 C605[E26F0000]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
00003548 C605[E26F0000]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);
0000354F C605[DE6F0000]0E       <1>         mov     byte [CHAR_HEIGHT], 14
                                <1>       ; }
00003556 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)
```

```
7193                                    <1>       ; /* set 0x43 INT pointer */
7194                                    <1>       ; write_word(0x0, 0x43*4, &vgafont8);
7195                                    <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
7196 00003557 C705[C2960100]-            <1>       mov    dword [VGA_INT43H], vgafont8
7196 0000355D [A8600100]                <1>
7197                                    <1>
7198                                    <1>       ; BL    screen rows code: 00H = user-specified (in DL)
7199                                    <1>         ;                         01H = 14 rows
7200                                    <1>         ;                         02H = 25 rows
7201                                    <1>         ;                         03H = 43 rows
7202                                    <1>         ; DL    (when BL=0) custom number of char rows on screen
7203                                    <1>
7204                                    <1>       ; switch (BL) {
7205                                    <1>       ; case 0:
7206                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7207                                    <1>       ;    break;
7208 00003561 20DB                      <1>       and    bl, bl
7209 00003563 7508                      <1>       jnz    short l_gfx_8_8c_1
7210 00003565 8815[E26F0000]            <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
7211 0000356B EB23                      <1>       jmp    short l_gfx_8_8c_4
7212                                    <1> l_gfx_8_8c_1:
7213                                    <1>       ; case 1:
7214                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7215                                    <1>       ;    break;
7216 0000356D FECB                      <1>       dec    bl
7217 0000356F 7509                      <1>       jnz    short l_gfx_8_8c_2
7218                                    <1>       ; bl = 1
7219 00003571 C605[E26F0000]0E          <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
7220 00003578 EB16                      <1>       jmp    short l_gfx_8_8c_4
7221                                    <1> l_gfx_8_8c_2:
7222 0000357A FECB                      <1>       dec    bl
7223 0000357C 740B                      <1>       jz     short l_gfx_8_8c_3 ; bl = 2
7224 0000357E FECB                      <1>       dec    bl
7225 00003580 750E                      <1>       jnz    short l_gfx_8_8c_4 ; bl > 3
7226                                    <1>       ; bl = 3
7227                                    <1>       ; case 3:
7228                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7229                                    <1>       ;    break;
7230 00003582 C605[E26F0000]2B          <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
7231                                    <1> l_gfx_8_8c_3:
7232                                    <1>       ; case 2:
7233                                    <1>       ; default:
7234                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7235                                    <1>       ;    break;
7236                                    <1>       ; bl = 2 or bl > 3
7237 00003589 C605[E26F0000]19          <1>       mov    byte [VGA_ROWS], 25  ; not 24 !
7238                                    <1>       ; }
7239                                    <1> l_gfx_8_8c_4:
7240                                    <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 8);
7241 00003590 C605[DE6F0000]08          <1>         mov     byte [CHAR_HEIGHT], 8
7242                                    <1>       ; }
7243 00003597 C3                        <1>       retn
7244                                    <1>
7245                                    <1> load_gfx_8_16_chars:
7246                                    <1>       ; 08/08/2016
7247                                    <1>       ; 10/07/2016
7248                                    <1>       ; Setup ROM 8x14 Font for Graphics Mode (VGA)
7249                                    <1>       ;
7250                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7251                                    <1>       ; vgabios-0.7a (2011)
7252                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7253                                    <1>       ; 'vgabios.c', 'biosfn_load_gfx_8_16_chars'
7254                                    <1>
7255                                    <1>       ; biosfn_load_gfx_8_16_chars (BL)
7256                                    <1>       ; /* set 0x43 INT pointer */
7257                                    <1>       ; write_word(0x0, 0x43*4, &vgafont16);
7258                                    <1>       ; write_word(0x0, 0x43*4+2, 0xC000);
7259 00003598 C705[C2960100]-            <1>       mov    dword [VGA_INT43H], vgafont16
7259 0000359E [A8760100]                <1>
7260                                    <1>
7261                                    <1>       ; BL    screen rows code: 00H = user-specified (in DL)
7262                                    <1>         ;                         01H = 14 rows
7263                                    <1>         ;                         02H = 25 rows
7264                                    <1>         ;                         03H = 43 rows
7265                                    <1>         ; DL    (when BL=0) custom number of char rows on screen
7266                                    <1>
7267                                    <1>       ; switch (BL) {
7268                                    <1>       ; case 0:
7269                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, DL-1);
7270                                    <1>       ;    break;
7271 000035A2 20DB                      <1>       and    bl, bl
7272 000035A4 7508                      <1>       jnz    short l_gfx_8_16c_1
7273 000035A6 8815[E26F0000]            <1>       mov    [VGA_ROWS], dl  ; not DL-1 !
7274 000035AC EB23                      <1>       jmp    short l_gfx_8_16c_4
7275                                    <1> l_gfx_8_16c_1:
7276                                    <1>       ; case 1:
7277                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 13);
7278                                    <1>       ;    break;
7279 000035AE FECB                      <1>       dec    bl
7280 000035B0 7509                      <1>       jnz    short l_gfx_8_16c_2
7281                                    <1>       ; bl = 1
7282 000035B2 C605[E26F0000]0E          <1>       mov    byte [VGA_ROWS], 14  ; not 13 !
7283 000035B9 EB16                      <1>       jmp    short l_gfx_8_16c_4
7284                                    <1> l_gfx_8_16c_2:
7285 000035BB FECB                      <1>       dec    bl
7286 000035BD 740B                      <1>       jz     short l_gfx_8_16c_3 ; bl = 2
7287 000035BF FECB                      <1>       dec    bl
7288 000035C1 750E                      <1>       jnz    short l_gfx_8_16c_4 ; bl > 3
7289                                    <1>       ; bl = 3
7290                                    <1>       ; case 3:
7291                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 42);
7292                                    <1>       ;    break;
7293 000035C3 C605[E26F0000]2B          <1>       mov    byte [VGA_ROWS], 43  ; not 42 !
7294                                    <1> l_gfx_8_16c_3:
7295                                    <1>       ; case 2:
```

```
7296                                    <1>       ; default:
7297                                    <1>       ;    write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, 24);
7298                                    <1>       ;    break;
7299                                    <1>       ; bl = 2 or bl > 3
7300 000035CA C605[E26F0000]19         <1>       mov   byte [VGA_ROWS], 25  ; not 24 !
7301                                    <1>       ; }
7302                                    <1> l_gfx_8_16c_4:
7303                                    <1>       ; write_byte(BIOSMEM_SEG, BIOSMEM_CHAR_HEIGHT, 16);
7304 000035D1 C605[DE6F0000]10         <1>         mov     byte [CHAR_HEIGHT], 16
7305                                    <1>       ; }
7306 000035D8 C3                       <1>       retn
7307                                    <1>
7308                                    <1> get_font_info:
7309                                    <1>       ; 08/01/2021 (TRDOS 386 v2.0.3)
7310                                    <1>       ; 19/09/2016
7311                                    <1>       ; 08/08/2016
7312                                    <1>       ; 10/07/2016
7313                                    <1>       ; Get Current Character Generator Info (VGA)
7314                                    <1>       ;
7315                                    <1>       ; derived from 'Plex86/Bochs VGABios' source code
7316                                    <1>       ; vgabios-0.7a (2011)
7317                                    <1>       ; by the LGPL VGABios developers Team (2001-2008)
7318                                    <1>       ; 'vgabios.c', 'biosfn_get_font_info'
7319                                    <1>
7320                                    <1>       ; Modified for TRDOS 386 !
7321                                    <1>       ;
7322                                    <1>       ; INPUT ->
7323                                    <1>       ;    AX = 1130h
7324                                    <1>       ;    BL = 0 -> Get info for current VGA font
7325                                    <1>       ;              (BH = unused)
7326                                    <1>       ;    19/09/2016
7327                                    <1>       ;    BL > 0 -> Get requested character font data
7328                                    <1>       ;       BL = 1 -> vgafont8
7329                                    <1>       ;        BL = 2 -> vgafont14
7330                                    <1>       ;       BL = 3 -> vgafont16
7331                                    <1>       ;     ;08/01/2021
7332                                    <1>       ;       BL = 4 -> user defined 8x8 font
7333                                    <1>       ;       BL = 5 -> user defined 8x14 font
7334                                    <1>       ;       BL = 6 -> user defined 8x16 font
7335                                    <1>       ;       BL > 6 -> Invalid function (for now!)
7336                                    <1>       ;       BH = ASCII code of the first character
7337                                    <1>       ;       ECX = Number of characters from the 1st char
7338                                    <1>       ;       ECX >= 256 -> All (256-BH) characters
7339                                    <1>       ;       ECX = 0 -> All characters (BH = unused)
7340                                    <1>       ;        EDX = User's Buffer Address
7341                                    <1>       ; OUTPUT ->
7342                                    <1>       ;    AL = height (scanlines), bytes per character
7343                                    <1>       ;    AH = screen rows
7344                                    <1>       ;    Byte 16-23 of EAX = number of columns
7345                                    <1>       ;    Byte 24-31 of EAX =
7346                                    <1>       ;        0 -> default font (not configured yet)
7347                                    <1>       ;        0FFh -> user defined font
7348                                    <1>       ;        14 = vgafont14
7349                                    <1>       ;         8 = vgafont8
7350                                    <1>       ;        16 = vgafont16
7351                                    <1>       ;   If BL input > 0 ->
7352                                    <1>       ;       EAX = Actual transfer count
7353                                    <1>       ;
7354 000035D9 20DB                     <1>       and   bl, bl
7355 000035DB 740F                     <1>       jz    short gfi_1
7356                                    <1>       ; invalid function (input)
7357                                    <1>       ; 08/01/2021
7358 000035DD 80FB04                   <1>       cmp   bl, 4
7359 000035E0 7263                     <1>       jb    short gfi_5
7360 000035E2 7441                     <1>       je    short gfi_3
7361 000035E4 80FB06                   <1>       cmp   bl, 6
7362 000035E7 744C                     <1>       je    short gfi_4
7363                                    <1>       ; bh = 5 or bh > 6
7364                                    <1> gfi_0:
7365 000035E9 31C0                     <1>       xor   eax, eax ; 0
7366 000035EB C3                       <1>       retn
7367                                    <1> gfi_1:
7368 000035EC A0[DE6F0000]             <1>       mov   al, [CHAR_HEIGHT]
7369 000035F1 8A25[E26F0000]           <1>       mov   ah, [VGA_ROWS]
7370 000035F7 C1E010                   <1>       shl   eax, 16
7371 000035FA A0[DC6F0000]             <1>       mov   al, [CRT_COLS]
7372 000035FF 8B0D[C2960100]           <1>       mov   ecx, [VGA_INT43H]
7373 00003605 21C9                     <1>       and   ecx, ecx
7374 00003607 7418                     <1>       jz    short gfi_2 ; 0 = default font
7375                                    <1>       ; 08/01/2021
7376 00003609 FECC                     <1>       dec   ah ; 0FFh
7377 0000360B 81F900400900             <1>       cmp   ecx, VGAFONT16USER
7378 00003611 740E                     <1>       je    short gfi_2
7379 00003613 81F900500900             <1>       cmp   ecx, VGAFONT8USER
7380 00003619 7406                     <1>       je    short gfi_2
7381 0000361B 8A25[DE6F0000]           <1>       mov   ah, [CHAR_HEIGHT] ; font size = height
7382                                    <1> gfi_2:
7383 00003621 C1C010                   <1>       rol   eax, 16
7384 00003624 C3                       <1>       retn
7385                                    <1> gfi_3:
7386                                    <1>       ; 08/01/2021
7387 00003625 F605[7E120300]08         <1>       test  byte [ufont], 08h ; 8x8 user font
7388 0000362C 74BB                     <1>       jz    short gfi_0 ; not loaded !
7389 0000362E BE00500900               <1>       mov   esi, VGAFONT8USER ; *
7390                                    <1>       ;mov  bl, 8
7391                                    <1>       ;jmp  short gfi_8
7392 00003633 EB4D                     <1>       jmp   short gfi_10
7393                                    <1> gfi_4:
7394                                    <1>       ; 08/01/2021
7395 00003635 F605[7E120300]10         <1>       test  byte [ufont], 10h ; 8x16 user font
7396 0000363C 74AB                     <1>       jz    short gfi_0 ; not loaded !
7397 0000363E BE00400900               <1>       mov   esi, VGAFONT16USER ; *
7398 00003643 EB15                     <1>       jmp   short gfi_7
7399                                    <1> gfi_5:
7400 00003645 80FB02                   <1>       cmp   bl, 2
```

```
7401 00003648 7233              <1>       jb     short gfi_9
7402 0000364A 7709              <1>       ja     short gfi_6
7403                            <1>       ;BL = 2 -> vgafont14
7404 0000364C BE[A8680100]      <1>       mov    esi, vgafont14 ; *
7405 00003651 B30E              <1>       mov    bl, 14
7406 00003653 EB07              <1>       jmp    short gfi_8
7407                            <1> gfi_6:
7408                            <1>       ;BL = 3 -> vgafont16
7409 00003655 BE[A8760100]      <1>       mov    esi, vgafont16 ; *
7410                            <1> gfi_7:
7411 0000365A B310              <1>       mov    bl, 16
7412                            <1> gfi_8:
7413 0000365C 89D7              <1>       mov    edi, edx ; **
7414 0000365E 09C9              <1>       or     ecx, ecx
7415 00003660 7424              <1>       jz     short gfi_11 ; all chars from the 00h
7416 00003662 88F8              <1>       mov    al, bh ; character index
7417 00003664 F6E3              <1>       mul    bl ; char index * char height/size
7418 00003666 0FB7D0            <1>       movzx  edx, ax
7419 00003669 01D6              <1>       add    esi, edx ; *
7420 0000366B 66BAFF00          <1>       mov    dx, 255
7421 0000366F 28FA              <1>       sub    dl, bh
7422 00003671 6642              <1>       inc    dx
7423 00003673 39D1              <1>       cmp    ecx, edx
7424 00003675 770F              <1>       ja     short gfi_11
7425 00003677 7412              <1>       je     short gfi_12
7426 00003679 89D1              <1>       mov    ecx, edx
7427 0000367B EB0E              <1>       jmp    short gfi_12
7428                            <1> gfi_9:
7429                            <1>       ;BL = 1 -> vgafont8
7430 0000367D BE[A8600100]      <1>       mov    esi, vgafont8 ; *
7431                            <1> gfi_10:
7432 00003682 B308              <1>       mov    bl, 8
7433 00003684 EBD6              <1>       jmp    short gfi_8
7434                            <1> gfi_11:
7435 00003686 B900010000        <1>       mov    ecx, 256
7436                            <1> gfi_12:
7437                            <1>       ; 08/01/2021
7438 0000368B 89C8              <1>       mov    eax, ecx ; character count
7439 0000368D 30FF              <1>       xor    bh, bh
7440 0000368F 66F7E3            <1>       mul    bx ; char count * char height/size
7441 00003692 89C1              <1>       mov    ecx, eax
7442                            <1>
7443                            <1>       ; ESI = source address in system space
7444                            <1>       ; EDI = user's buffer address
7445                            <1>       ; ECX = transfer (byte) count
7446 00003694 E8AEE40000        <1>       call   transfer_to_user_buffer
7447 00003699 89C8              <1>       mov    eax, ecx ; actual transfer count
7448 0000369B C3                <1>       retn
7449                            <1>
7450                            <1> vga_pal_funcs:
7451                            <1>       ; 10/08/2016
7452                            <1>       ; VGA Palette functions
7453                            <1>       ;
7454                            <1>       ; derived from 'Plex86/Bochs VGABios' source code
7455                            <1>       ; vgabios-0.7a (2011)
7456                            <1>       ; by the LGPL VGABios developers Team (2001-2008)
7457                            <1>       ; 'vgabios.c', 'vgarom.asm'
7458                            <1>
7459 0000369C 3C00              <1>       cmp    al, 0
7460 0000369E 0F848F000000      <1>         je      set_single_palette_reg
7461                            <1> vga_palf_1001:
7462 000036A4 3C01              <1>       cmp    al, 1
7463 000036A6 0F84B4000000      <1>         je      set_overscan_border_color
7464                            <1> vga_palf_1002:
7465 000036AC 3C02              <1>       cmp    al, 2
7466 000036AE 0F84B0000000      <1>         je      set_all_palette_reg
7467                            <1> vga_palf_1003:
7468 000036B4 3C03              <1>       cmp    al, 3
7469 000036B6 0F84E8000000      <1>         je      toggle_intensity
7470                            <1> vga_palf_1007:
7471 000036BC 3C07              <1>       cmp    al, 7
7472 000036BE 0F840D010000      <1>         je      get_single_palette_reg
7473 000036C4 7266              <1>       jb     short vga_palf_unknown
7474                            <1> vga_palf_1008:
7475 000036C6 3C08              <1>       cmp    al, 8
7476 000036C8 0F8437010000      <1>         je      read_overscan_border_color
7477                            <1> vga_palf_1009:
7478 000036CE 3C09              <1>       cmp    al, 9
7479 000036D0 0F8433010000      <1>       je     get_all_palette_reg
7480                            <1> vga_palf_1010:
7481 000036D6 3C10              <1>       cmp    al, 10h
7482 000036D8 0F8487010000      <1>       je     set_single_dac_reg
7483 000036DE 724C              <1>       jb     short vga_palf_unknown
7484                            <1> vga_palf_1012:
7485 000036E0 3C12              <1>       cmp    al, 12h
7486 000036E2 0F8498010000      <1>         je      set_all_dac_reg
7487 000036E8 7242              <1>       jb     short vga_palf_unknown
7488                            <1> vga_palf_1013:
7489 000036EA 3C13              <1>       cmp    al, 13h
7490 000036EC 0F84CC010000      <1>         je      select_video_dac_color_page
7491                            <1> vga_palf_1015:
7492 000036F2 3C15              <1>       cmp    al, 15h
7493 000036F4 0F8412020000      <1>       je     read_single_dac_reg
7494 000036FA 7230              <1>       jb     short vga_palf_unknown
7495                            <1> vga_palf_1017:
7496 000036FC 3C17              <1>       cmp    al, 17h
7497 000036FE 0F8428020000      <1>         je      read_all_dac_reg
7498 00003704 7226              <1>       jb     short vga_palf_unknown
7499                            <1> vga_palf_1018:
7500 00003706 3C18              <1>       cmp    al, 18h
7501 00003708 0F845E020000      <1>         je      set_pel_mask
7502                            <1> vga_palf_1019:
7503 0000370E 3C19              <1>       cmp    al, 19h
7504 00003710 0F8462020000      <1>         je      read_pel_mask
7505                            <1> vga_palf_101A:
```

```
7506 00003716 3C1A              <1>       cmp    al, 1Ah
7507 00003718 0F8468020000      <1>         je      read_video_dac_state
7508                            <1> vga_palf_101B:
7509 0000371E 3C1B              <1>       cmp    al, 1Bh
7510                            <1>       ;jne   short vga_palf_unknown
7511 00003720 770A              <1>       ja     short vga_palf_unknown
7512                            <1>
7513 00003722 E8C3F5FFFF        <1>       call   gray_scale_summing
7514 00003727 E933E4FFFF        <1>         jmp     VIDEO_RETURN
7515                            <1>
7516                            <1> vga_palf_unknown:
7517 0000372C 29C0              <1>       sub    eax, eax ; 0 = invalid function
7518 0000372E E931E4FFFF        <1>         jmp     _video_return
7519                            <1>
7520                            <1> set_single_palette_reg:
7521                            <1>       ; 10/08/2016
7522                            <1>       ; Set One Palette Register
7523                            <1>       ; BL = register number to set
7524                            <1>       ;     (a 4-bit attribute nibble: 00h-0Fh)
7525                            <1>       ; BH = 6-bit RGB color to display
7526                            <1>       ;      for that attribute
7527                            <1>
7528 00003733 80FB14            <1>       cmp    bl, 14h
7529                            <1>       ;ja    short no_actl_reg1
7530 00003736 0F8723E4FFFF      <1>       ja     VIDEO_RETURN
7531 0000373C 6650              <1>       push   ax
7532 0000373E 6652              <1>       push   dx
7533 00003740 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7534 00003744 EC                <1>       in     al, dx
7535 00003745 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7536 00003749 88D8              <1>       mov    al, bl
7537 0000374B EE                <1>       out    dx, al
7538 0000374C 88F8              <1>       mov    al, bh
7539 0000374E EE                <1>       out    dx, al
7540 0000374F B020              <1>       mov    al, 20h
7541 00003751 EE                <1>       out    dx, al
7542                            <1>       ; ifdef VBOX
7543 00003752 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7544 00003756 EC                <1>       in     al, dx
7545                            <1>       ; endif ; VBOX
7546 00003757 665A              <1>       pop    dx
7547 00003759 6658              <1>       pop    ax
7548                            <1> ;no_actl_reg1:
7549 0000375B E9FFE3FFFF        <1>       jmp    VIDEO_RETURN
7550                            <1>
7551                            <1> set_overscan_border_color:
7552                            <1>       ; 10/08/2016
7553                            <1>       ; Set Overscan/Border Color Register
7554                            <1>       ; BH = 6-bit RGB color to display
7555                            <1>       ;      for that attribute
7556                            <1>
7557 00003760 B311              <1>       mov    bl, 11h
7558 00003762 EBCF              <1>       jmp    short set_single_palette_reg
7559                            <1>
7560                            <1> set_all_palette_reg:
7561                            <1>       ; 10/08/2016
7562                            <1>       ; Set All Palette Registers and Overscan
7563                            <1>       ; EDX = Address of 17 bytes;
7564                            <1>       ; an rgbRGB value for each of 16 palette
7565                            <1>       ; registers plus one for the border.
7566                            <1>
7567 00003764 89D6              <1>       mov    esi, edx ; user buffer
7568 00003766 B911000000        <1>       mov    ecx, 17
7569 0000376B 89E7              <1>       mov    edi, esp
7570 0000376D 83EC14            <1>       sub    esp, 20
7571 00003770 E81CE40000        <1>       call   transfer_from_user_buffer
7572                            <1>       ;jc    VIDEO_RETURN
7573                            <1>
7574 00003775 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7575 00003779 EC                <1>       in     al, dx
7576 0000377A B100              <1>       mov    cl, 0
7577 0000377C 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7578                            <1> set_palette_loop:
7579 00003780 88C8              <1>       mov    al, cl
7580 00003782 EE                <1>       out    dx, al
7581 00003783 8A07              <1>       mov    al, [edi]
7582 00003785 EE                <1>       out    dx, al
7583 00003786 47                <1>       inc    edi
7584 00003787 FEC1              <1>       inc    cl
7585 00003789 80F910            <1>       cmp    cl, 10h
7586 0000378C 75F2              <1>       jne    short set_palette_loop
7587 0000378E B011              <1>       mov    al, 11h
7588 00003790 EE                <1>       out    dx, al
7589 00003791 8A07              <1>       mov    al, [edi]
7590 00003793 EE                <1>       out    dx, al
7591 00003794 B020              <1>       mov    al, 20h
7592 00003796 EE                <1>       out    dx, al
7593                            <1>       ; ifdef VBOX
7594 00003797 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7595 0000379B EC                <1>       in     al, dx
7596                            <1>       ; endif ; VBOX
7597 0000379C 83C414            <1>       add    esp, 20
7598 0000379F E9BBE3FFFF        <1>       jmp    VIDEO_RETURN
7599                            <1>
7600                            <1> toggle_intensity:
7601                            <1>       ; 10/08/2016
7602                            <1>       ; Select Foreground Blink or Bold Background
7603                            <1>       ; BL = 00h = enable bold backgrounds
7604                            <1>       ;          (16 background colors)
7605                            <1>         ;      01h = enable blinking foreground
7606                            <1>       ;          (8 background colors)
7607                            <1>
7608 000037A4 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7609 000037A8 EC                <1>       in     al, dx
7610 000037A9 66BAC003          <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
```

```
7611 000037AD B010                <1>       mov    al, 10h
7612 000037AF EE                  <1>       out    dx, al
7613 000037B0 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7614 000037B4 EC                  <1>       in     al, dx
7615 000037B5 24F7                <1>       and    al, 0F7h
7616 000037B7 80E301              <1>       and    bl, 01h
7617 000037BA C0E303              <1>       shl    bl, 3
7618 000037BD 08D8                <1>       or     al, bl
7619 000037BF 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7620 000037C3 EE                  <1>       out    dx, al
7621 000037C4 B020                <1>       mov    al, 20h
7622 000037C6 EE                  <1>       out    dx, al
7623                              <1>       ; ifdef VBOX
7624 000037C7 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7625 000037CB EC                  <1>       in     al, dx
7626                              <1>       ; endif ; VBOX
7627 000037CC E98EE3FFFF          <1>       jmp    VIDEO_RETURN
7628                              <1>
7629                              <1> get_single_palette_reg:
7630                              <1>       ; 10/08/2016
7631                              <1>       ; Read One Palette Register
7632                              <1>         ; INPUT:
7633                              <1>       ; BL = Palette register to read (00h-0Fh)
7634                              <1>       ; OUTPUT:
7635                              <1>       ; BH = Current rgbRGB value of specified register
7636                              <1>       ;      for that attribute
7637                              <1>
7638 000037D1 80FB14              <1>       cmp    bl, 14h
7639                              <1>       ;ja    short no_actl_reg2
7640 000037D4 0F8785E3FFFF        <1>       ja     VIDEO_RETURN
7641                              <1>
7642 000037DA 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7643 000037DE EC                  <1>       in     al, dx
7644 000037DF 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7645 000037E3 88D8                <1>       mov    al, bl
7646 000037E5 EE                  <1>       out    dx, al
7647 000037E6 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7648 000037EA EC                  <1>       in     al, dx
7649 000037EB 8844240D            <1>       mov    [esp+13], al ; bh
7650 000037EF 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7651 000037F3 EC                  <1>       in     al, dx
7652 000037F4 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7653 000037F8 B020                <1>       mov    al, 20h
7654 000037FA EE                  <1>       out    dx, al
7655                              <1>       ; ifdef VBOX
7656 000037FB 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7657 000037FF EC                  <1>       in     al, dx
7658                              <1>       ; endif ; VBOX
7659 00003800 E95AE3FFFF          <1>       jmp    VIDEO_RETURN
7660                              <1>
7661                              <1> read_overscan_border_color:
7662                              <1>       ; 10/08/2016
7663                              <1>       ; Read Overscan Register
7664                              <1>       ; OUTPUT:
7665                              <1>       ; BH = current rgbRGB value
7666                              <1>       ;      of the overscan/border register
7667                              <1>
7668 00003805 B311                <1>       mov    bl, 11h
7669 00003807 EBC8                <1>       jmp    short get_single_palette_reg
7670                              <1>
7671                              <1> get_all_palette_reg:
7672                              <1>       ; 10/08/2016
7673                              <1>       ; Read All Palette Registers
7674                              <1>       ; EDX = Address of 17-byte buffer
7675                              <1>       ;      to receive data
7676                              <1>
7677 00003809 89D7                <1>       mov    edi, edx
7678 0000380B 89E3                <1>       mov    ebx, esp
7679 0000380D 89DE                <1>       mov    esi, ebx
7680 0000380F 83EC14              <1>       sub    esp, 20
7681                              <1>
7682 00003812 B100                <1>       mov    cl, 0
7683                              <1> get_palette_loop:
7684 00003814 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7685 00003818 EC                  <1>       in     al, dx
7686 00003819 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7687 0000381D 88C8                <1>       mov    al, cl
7688 0000381F EE                  <1>       out    dx, al
7689 00003820 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7690 00003824 EC                  <1>       in     al, dx
7691 00003825 8803                <1>       mov    [ebx], al
7692 00003827 43                  <1>       inc    ebx
7693 00003828 FEC1                <1>       inc    cl
7694 0000382A 80F910              <1>       cmp    cl, 10h
7695 0000382D 75E5                <1>       jne    short get_palette_loop
7696 0000382F 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7697 00003833 EC                  <1>       in     al, dx
7698 00003834 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7699 00003838 B011                <1>       mov    al, 11h
7700 0000383A EE                  <1>       out    dx, al
7701 0000383B 66BAC103            <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7702 0000383F EC                  <1>       in     al, dx
7703 00003840 8803                <1>       mov    [ebx], al
7704 00003842 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7705 00003846 EC                  <1>       in     al, dx
7706 00003847 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7707 0000384B B020                <1>       mov    al, 20h
7708 0000384D EE                  <1>       out    dx, al
7709                              <1>       ; ifdef VBOX
7710 0000384E 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7711 00003852 EC                  <1>       in     al, dx
7712                              <1>       ; endif ; VBOX
7713                              <1>
7714 00003853 B911000000          <1>       mov    ecx, 17 ; transfer (byte) count
7715                              <1>       ; ESI = source address in system space
```

```
7716                                    <1>       ; EDI = user's buffer address
7717 00003858 E8EAE20000                <1>       call   transfer_to_user_buffer
7718                                    <1>
7719 0000385D 83C414                    <1>       add    esp, 20
7720 00003860 E9FAE2FFFF                <1>       jmp    VIDEO_RETURN
7721                                    <1>
7722                                    <1> set_single_dac_reg:
7723                                    <1>       ; 10/08/2016
7724                                    <1>       ; Set One DAC Color Register
7725                                    <1>       ; BX = color register to set (0-255)
7726                                    <1>         ; CH = green value (00h-3Fh)
7727                                    <1>         ; CL = blue value  (00h-3Fh)
7728                                    <1>         ; DH = red value   (00h-3Fh)
7729                                    <1>
7730 00003865 6652                      <1>       push   dx
7731 00003867 66BAC803                  <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
7732 0000386B 88D8                      <1>       mov    al, bl
7733 0000386D EE                        <1>       out    dx, al
7734                                    <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
7735 0000386E 6642                      <1>       inc    dx
7736 00003870 6658                      <1>       pop    ax
7737 00003872 88E0                      <1>       mov    al, ah
7738 00003874 EE                        <1>       out    dx, al
7739 00003875 88E8                      <1>       mov    al, ch
7740 00003877 EE                        <1>       out    dx, al
7741 00003878 88C8                      <1>       mov    al, cl
7742 0000387A EE                        <1>       out    dx, al
7743 0000387B E9DFE2FFFF                <1>       jmp    VIDEO_RETURN
7744                                    <1>
7745                                    <1> set_all_dac_reg:
7746                                    <1>       ; 12/08/2016
7747                                    <1>       ; 11/08/2016
7748                                    <1>       ; 10/08/2016
7749                                    <1>       ; Set a Block of DAC Color Register
7750                                    <1>       ; BX = first DAC register to set (0-00FFh)
7751                                    <1>       ; ECX = number of registers to set (0-00FFh)
7752                                    <1>       ; EDX = addr of a table of R,G,B values
7753                                    <1>       ;      (it will be CX*3 bytes long)
7754                                    <1>
7755 00003880 89D6                      <1>       mov    esi, edx ; user buffer
7756 00003882 89CA                      <1>       mov    edx, ecx
7757 00003884 66D1E1                    <1>       shl    cx, 1 ; *2
7758 00003887 01D1                      <1>       add    ecx, edx ; ecx = 3*ecx
7759 00003889 89E5                      <1>       mov    ebp, esp
7760 0000388B 89EF                      <1>       mov    edi, ebp
7761 0000388D 29CF                      <1>       sub    edi, ecx
7762 0000388F 6683E7FC                  <1>       and    di, 0FFFCh ; (dword alignment)
7763 00003893 89FC                      <1>       mov    esp, edi
7764 00003895 E8F7E20000                <1>       call   transfer_from_user_buffer
7765                                    <1>       ;jc    VIDEO_RETURN
7766                                    <1>
7767 0000389A 89D1                      <1>       mov    ecx, edx
7768 0000389C 66BAC803                  <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
7769 000038A0 88D8                      <1>       mov    al, bl
7770 000038A2 EE                        <1>       out    dx, al
7771 000038A3 66BAC903                  <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
7772                                    <1> set_dac_loop:
7773 000038A7 8A07                      <1>       mov    al, [edi]
7774 000038A9 EE                        <1>       out    dx, al
7775 000038AA 47                        <1>       inc    edi
7776 000038AB 8A07                      <1>       mov    al, [edi]
7777 000038AD EE                        <1>       out    dx, al
7778 000038AE 47                        <1>       inc    edi
7779 000038AF 8A07                      <1>       mov    al, [edi]
7780 000038B1 EE                        <1>       out    dx, al
7781 000038B2 47                        <1>       inc    edi
7782 000038B3 6649                      <1>       dec    cx
7783 000038B5 75F0                      <1>       jnz    short set_dac_loop
7784 000038B7 89EC                      <1>       mov    esp, ebp
7785 000038B9 E9A1E2FFFF                <1>       jmp    VIDEO_RETURN
7786                                    <1>
7787                                    <1> select_video_dac_color_page:
7788                                    <1>       ; 10/08/2016
7789                                    <1>       ; DAC Color Paging Functions
7790                                    <1>       ; BL = 00H = select color paging mode
7791                                    <1>         ;       BH = paging mode
7792                                    <1>         ;            00h = 4 blocks of 64 registers
7793                                    <1>         ;            01h = 16 blocks of 16 registers
7794                                    <1>       ; BL = 01H = activate color page
7795                                    <1>         ;       BH = DAC color page number
7796                                    <1>         ;            00h-03h (4-page/64-reg mode)
7797                                    <1>         ;            00h-0Fh (16-page/16-reg mode)
7798                                    <1>
7799 000038BE 66BADA03                  <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7800 000038C2 EC                        <1>       in     al, dx
7801 000038C3 66BAC003                  <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7802 000038C7 B010                      <1>       mov    al, 10h
7803 000038C9 EE                        <1>       out    dx, al
7804 000038CA 66BAC103                  <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
7805 000038CE EC                        <1>       in     al, dx
7806 000038CF 80E301                    <1>       and    bl, 01h
7807 000038D2 750E                      <1>       jnz    short set_dac_page
7808 000038D4 247F                      <1>       and    al, 07Fh
7809 000038D6 C0E707                    <1>       shl    bh, 7
7810 000038D9 08F8                      <1>       or     al, bh
7811 000038DB 66BAC003                  <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7812 000038DF EE                        <1>       out    dx, al
7813 000038E0 EB1D                      <1>       jmp    short set_actl_normal
7814                                    <1> set_dac_page:
7815 000038E2 6650                      <1>       push   ax
7816 000038E4 66BADA03                  <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7817 000038E8 EC                        <1>       in     al, dx
7818 000038E9 66BAC003                  <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
7819 000038ED B014                      <1>       mov    al, 14h
7820 000038EF EE                        <1>       out    dx, al
```

```
7821 000038F0 6658              <1>       pop    ax
7822 000038F2 2480              <1>       and    al, 80h
7823 000038F4 7503              <1>       jnz    short set_dac_16_page
7824 000038F6 C0E702            <1>       shl    bh, 2
7825                            <1> set_dac_16_page:
7826 000038F9 80E70F            <1>       and    bh, 0Fh
7827 000038FC 88F8              <1>       mov    al, bh
7828 000038FE EE                <1>       out    dx, al
7829                            <1> set_actl_normal:
7830 000038FF B020              <1>       mov    al, 20h
7831 00003901 EE                <1>       out    dx, al
7832                            <1>       ; ifdef VBOX
7833 00003902 66BADA03          <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
7834 00003906 EC                <1>       in     al, dx
7835                            <1>       ; endif ; VBOX
7836 00003907 E953E2FFFF        <1>       jmp    VIDEO_RETURN
7837                            <1>
7838                            <1> read_single_dac_reg:
7839                            <1>       ; 10/08/2016
7840                            <1>       ; Read One DAC Color Register
7841                            <1>       ; INPUT:
7842                            <1>       ; BX = color register to read (0-255)
7843                            <1>       ; OUTPUT:
7844                            <1>       ; CH = green value (00h-3Fh)
7845                            <1>         ; CL = blue value  (00h-3Fh)
7846                            <1>         ; DH = red value   (00h-3Fh)
7847                            <1>
7848 0000390C 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
7849 00003910 88D8              <1>       mov    al, bl
7850 00003912 EE                <1>       out    dx, al
7851 00003913 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
7852 00003917 EC                <1>       in     al, dx
7853 00003918 88442415          <1>       mov    [esp+21], al ; dh
7854 0000391C EC                <1>       in     al, dx
7855 0000391D 88C5              <1>       mov    ch, al
7856 0000391F EC                <1>       in     al, dx
7857 00003920 88C1              <1>       mov    cl, al
7858 00003922 66894C2410        <1>       mov    [esp+16], cx ; cx
7859 00003927 E933E2FFFF        <1>       jmp    VIDEO_RETURN
7860                            <1>
7861                            <1> read_all_dac_reg:
7862                            <1>       ; 12/08/2016
7863                            <1>       ; 11/08/2016
7864                            <1>       ; 10/08/2016
7865                            <1>       ; Read a Block of DAC Color Registers
7866                            <1>         ; BX = first DAC register to read (0-00FFh)
7867                            <1>         ; ECX = number of registers to read (0-00FFh)
7868                            <1>         ; EDX = addr of a buffer to hold R,G,B values
7869                            <1>       ;      (CX*3 bytes long)
7870                            <1>
7871 0000392C 89D7              <1>       mov    edi, edx ; user buffer
7872 0000392E 89CA              <1>       mov    edx, ecx
7873 00003930 66D1E2            <1>       shl    dx, 1 ; *2
7874 00003933 01CA              <1>       add    edx, ecx ; edx = 3*ecx
7875 00003935 89E5              <1>       mov    ebp, esp
7876 00003937 89EE              <1>       mov    esi, ebp
7877 00003939 29D6              <1>       sub    esi, edx
7878 0000393B 6683E6FC          <1>       and    si, 0FFFCh ; (dword alignment)
7879 0000393F 89F4              <1>       mov    esp, esi
7880 00003941 52                <1>       push   edx ; 3*ecx
7881 00003942 66BAC703          <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
7882 00003946 88D8              <1>       mov    al, bl
7883 00003948 EE                <1>       out    dx, al
7884 00003949 66BAC903          <1>       mov    dx, 3C9h ; VGAREG_DAC_DATA
7885 0000394D 89F3              <1>       mov    ebx, esi
7886                            <1> read_dac_loop:
7887 0000394F EC                <1>       in     al, dx
7888 00003950 8803              <1>       mov    [ebx], al
7889 00003952 43                <1>       inc    ebx
7890 00003953 EC                <1>       in     al, dx
7891 00003954 8803              <1>       mov    [ebx], al
7892 00003956 43                <1>       inc    ebx
7893 00003957 EC                <1>       in     al, dx
7894 00003958 8803              <1>       mov    [ebx], al
7895 0000395A 43                <1>       inc    ebx
7896 0000395B 6649              <1>       dec    cx
7897 0000395D 75F0              <1>       jnz    short read_dac_loop
7898 0000395F 59                <1>       pop    ecx ; 3*ecx
7899                            <1>       ; ECX = transfer (byte) count
7900                            <1>       ; ESI = source address in system space
7901                            <1>       ; EDI = user's buffer address
7902 00003960 E8E2E10000        <1>       call   transfer_to_user_buffer
7903 00003965 89EC              <1>       mov    esp, ebp
7904 00003967 E9F3E1FFFF        <1>       jmp    VIDEO_RETURN
7905                            <1>
7906                            <1> set_pel_mask:
7907                            <1>       ; 10/08/2016
7908                            <1>       ; BL = mask value
7909 0000396C 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
7910 00003970 88D8              <1>       mov    al, bl
7911 00003972 EE                <1>       out    dx, al
7912 00003973 E9E7E1FFFF        <1>       jmp    VIDEO_RETURN
7913                            <1>
7914                            <1> read_pel_mask:
7915                            <1>       ; 10/08/2016
7916                            <1>       ; Output: BL = mask value
7917 00003978 66BAC603          <1>       mov    dx, 3C6h ; VGAREG_PEL_MASK
7918 0000397C EC                <1>       in     al, dx
7919 0000397D 8844240C          <1>       mov    [esp+12], al ; bl
7920 00003981 E9D9E1FFFF        <1>       jmp    VIDEO_RETURN
7921                            <1>
7922                            <1> read_video_dac_state:
7923                            <1>       ; 10/08/2016
7924                            <1>       ; Query DAC Color Paging State
7925                            <1>       ; Output:
```

```
                                <1>       ; BH = current active DAC color page
                                <1>         ; BL = current active DAC paging mode
                                <1>
00003986 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
0000398A EC                     <1>       in     al, dx
0000398B 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
0000398F B010                   <1>       mov    al, 10h
00003991 EE                     <1>       out    dx, al
00003992 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
00003996 EC                     <1>       in     al, dx
00003997 88C3                   <1>       mov    bl, al
00003999 C0EB07                 <1>       shr    bl, 7
0000399C 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000039A0 EC                     <1>       in     al, dx
000039A1 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000039A5 B014                   <1>       mov    al, 14h
000039A7 EE                     <1>       out    dx, al
000039A8 66BAC103               <1>       mov    dx, 3C1h ; VGAREG_ACTL_READ_DATA
000039AC EC                     <1>       in     al, dx
000039AD 88C7                   <1>       mov    bh, al
000039AF 80E70F                 <1>       and    bh, 0Fh
000039B2 F6C301                 <1>       test   bl, 01
000039B5 7503                   <1>       jnz    short get_dac_16_page
000039B7 C0EF02                 <1>       shr    bh, 2
                                <1> get_dac_16_page:
000039BA 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000039BE EC                     <1>       in     al, dx
000039BF 66BAC003               <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
000039C3 B020                   <1>       mov    al, 20h
000039C5 EE                     <1>       out    dx, al
                                <1>       ; ifdef VBOX
000039C6 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
000039CA EC                     <1>       in     al, dx
                                <1>       ; endif ; VBOX
000039CB 66895C240C             <1>       mov    [esp+12], bx ; bx
000039D0 E98AE1FFFF             <1>       jmp    VIDEO_RETURN
                                <1>
                                <1> ; 23/11/2020 - TRDOS 386 v2.0.3
                                <1> ; VBE 2 BOCHS/QEMU emulator extensions
                                <1> ;     for TRDOS 386 v2 kernel (video bios)
                                <1>
                                <1> ; BOCH/QEMU VBE2 VGA BIOS code
                                <1> ;     by Jeroen Janssen (2002)
                                <1> ;     by Volker Rupper (2003-2020)
                                <1> ; vbe.c (02/01/2020)
                                <1>
                                <1> ; vbe.h (02/01/2020)
                                <1>
                                <1> VBE_DISPI_BANK_ADDRESS    equ    0A0000h
                                <1> VBE_DISPI_BANK_SIZE_KB    equ    64
                                <1>
                                <1> VBE_DISPI_MAX_XRES equ    2560
                                <1> VBE_DISPI_MAX_YRES equ    1600
                                <1>
                                <1> VBE_DISPI_IOPORT_INDEX    equ    01CEh
                                <1> VBE_DISPI_IOPORT_DATA     equ    01CFh
                                <1>
                                <1> VBE_DISPI_INDEX_ID equ    00h
                                <1> VBE_DISPI_INDEX_XRES      equ    01h
                                <1> VBE_DISPI_INDEX_YRES      equ    02h
                                <1> VBE_DISPI_INDEX_BPP       equ    03h
                                <1> VBE_DISPI_INDEX_ENABLE    equ    04h
                                <1> VBE_DISPI_INDEX_BANK      equ    05h
                                <1> VBE_DISPI_INDEX_VIRT_WIDTH equ  06h
                                <1> VBE_DISPI_INDEX_VIRT_HEIGHT equ 07h
                                <1> VBE_DISPI_INDEX_X_OFFSET equ    08h
                                <1> VBE_DISPI_INDEX_Y_OFFSET equ    09h
                                <1> VBE_DISPI_INDEX_VIDEO_MEMORY_64K equ 0Ah
                                <1> VBE_DISPI_INDEX_DDC       equ    0Bh
                                <1>
                                <1> VBE_DISPI_ID0             equ    0B0C0h
                                <1> VBE_DISPI_ID1             equ    0B0C1h
                                <1> VBE_DISPI_ID2             equ    0B0C2h
                                <1> VBE_DISPI_ID3             equ    0B0C3h
                                <1> VBE_DISPI_ID4             equ    0B0C4h
                                <1> VBE_DISPI_ID5             equ    0B0C5h
                                <1>
                                <1> VBE_DISPI_DISABLED equ    00h
                                <1> VBE_DISPI_ENABLED  equ    01h
                                <1> VBE_DISPI_GETCAPS  equ    02h
                                <1> VBE_DISPI_8BIT_DAC equ    20h
                                <1> VBE_DISPI_LFB_ENABLED     equ    40h
                                <1> VBE_DISPI_NOCLEARMEM      equ    80h
                                <1>
                                <1> VBE_DISPI_LFB_PHYSICAL_ADDRESS equ 0E0000000h
                                <1>
                                <1> ; ***
                                <1>
                                <1> ;// VBE Return Status Info
                                <1> ;// AL
                                <1> VBE_RETURN_STATUS_SUPPORTED     equ    4Fh
                                <1> VBE_RETURN_STATUS_UNSUPPORTED   equ    00h
                                <1> ;// AH
                                <1> VBE_RETURN_STATUS_SUCCESSFULL   equ    00h
                                <1> VBE_RETURN_STATUS_FAILED  equ    01h
                                <1> VBE_RETURN_STATUS_NOT_SUPPORTED equ    02h
                                <1> VBE_RETURN_STATUS_INVALID equ    03h
                                <1>
                                <1> ;// VBE Mode Numbers
                                <1>
                                <1> VBE_MODE_VESA_DEFINED           equ    0100h
                                <1> VBE_MODE_REFRESH_RATE_USE_CRTC  equ    0800h
                                <1> VBE_MODE_LINEAR_FRAME_BUFFER    equ    4000h
                                <1> VBE_MODE_PRESERVE_DISPLAY_MEMORY equ   8000h
                                <1>
```

```
                        <1> ;// Mode Attributes
                        <1>
                        <1> VBE_MODE_ATTRIBUTE_SUPPORTED              equ 0001h
                        <1> VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE equ      0002h
                        <1> VBE_MODE_ATTRIBUTE_COLOR_MODE             equ 0008h
                        <1> VBE_MODE_ATTRIBUTE_GRAPHICS_MODE    equ 0010h
                        <1> VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE equ     0080h
                        <1> VBE_MODE_ATTRIBUTE_DOUBLE_SCAN_MODE       equ 0100h
                        <1> VBE_MODE_ATTRIBUTE_INTERLACE_MODE         equ 0200h
                        <1>
                        <1> ;// Window attributes
                        <1>
                        <1> VBE_WINDOW_ATTRIBUTE_RELOCATABLE equ   01h
                        <1> VBE_WINDOW_ATTRIBUTE_READABLE    equ   02h
                        <1> VBE_WINDOW_ATTRIBUTE_WRITEABLE   equ   04h
                        <1>
                        <1> ;/* Video memory */
                        <1> VGAMEM_GRAPH equ 0A000h
                        <1> VGAMEM_CTEXT equ 0B800h
                        <1> ;VGAMEM_MTEXT equ 0B000h
                        <1>
                        <1> ;// Memory model
                        <1>
                        <1> ;VBE_MEMORYMODEL_TEXT_MODE       equ    00h
                        <1> ;VBE_MEMORYMODEL_CGA_GRAPHICS    equ    01h
                        <1> ;VBE_MEMORYMODEL_PLANAR          equ    03h
                        <1> VBE_MEMORYMODEL_PACKED_PIXEL     equ    04h
                        <1> ;VBE_MEMORYMODEL_NON_CHAIN_4_256 equ   05h
                        <1> VBE_MEMORYMODEL_DIRECT_COLOR     equ    06h
                        <1> ;VBE_MEMORYMODEL_YUV             equ    07h
                        <1>
                        <1> ;// DirectColorModeInfo
                        <1>
                        <1> ;VBE_DIRECTCOLOR_COLOR_RAMP_PROGRAMMABLE equ 01h
                        <1> VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE equ 02h
                        <1>
                        <1> VBE_DISPI_TOTAL_VIDEO_MEMORY_MB equ 16
                        <1>
                        <1> ; 24/11/2020
                        <1> ; vbe.c
                        <1>
                        <1> %if 1
                        <1>
                        <1> _vbe_biosfn_return_mode_info:
                        <1>       ; 15/12/2020
                        <1>       ; 12/12/2020
                        <1>       ; Return VBE Mode Information
                        <1>       ; (call from 'sysvideo')
                        <1>       ;
                        <1>       ; Input:
                        <1>       ;     cx = video (bios) mode
                        <1>       ; Output:
                        <1>       ;     cf = 0 -> (successful)
                        <1>       ;        MODE_INFO_LIST addr contains MODEINFO
                        <1>       ;     cf = 1 -> error
                        <1>       ;
                        <1>       ; Modified registers: eax, edx, edi
                        <1>       ;
                        <1>
                        <1>       ; pushes for subroutine stack pops compatibility
                        <1>
                        <1>       ;push  ds  ; *
                        <1>       ;push  es  ; **
                        <1>
 000039D5 55            <1>       push   ebp ; ***
 000039D6 56            <1>       push   esi ; ****
                        <1>
 000039D7 31FF          <1>       xor    edi, edi  ; mov edi, 0
                        <1>
 000039D9 803D[5C090000]03  <1>   cmp    byte [vbe3], 3
 000039E0 7221          <1>       jb     short _vbe_rmi_1
                        <1>
                        <1>       ;sub   edi, edi  ; 0 = kernel call (sign)
                        <1>       ;      ; no transfer to user's buffer
                        <1>
                        <1>       ; cx = Video mode (for 4F01h, with LFB flag)
                        <1>
 000039E2 66B8014F      <1>       mov    ax, 4F01h
                        <1>
 000039E6 E820DFFFFF    <1>       call   _vbe3_pmfn_return_mode_info
                        <1>
 000039EB 6683F84F      <1>       cmp    ax, 004Fh
 000039EF 7533          <1>       jne    short _vbe_rmi_2 ; fail
                        <1>
                        <1>       ; 15/12/2020
                        <1>       ; cx = vbe video mode
 000039F1 80E501        <1>       and    ch, 01h ; clear LFB flag
 000039F4 BEFE7B0900    <1>       mov    esi, VBE3MODEINFOBLOCK - 2
 000039F9 66890E        <1>       mov    [esi], cx ; MODEINFO.mode
 000039FC E8A6000000    <1>       call   set_lfbinfo_table
 00003A01 EB22          <1>       jmp    short _vbe_rmi_3  ; cf = 0
                        <1> _vbe_rmi_1:
 00003A03 803D[5C090000]02  <1>   cmp    byte [vbe3], 2
 00003A0A 7219          <1>       jb     short _vbe_rmi_3 ; cf = 1
 00003A0C A0[5D090000]  <1>       mov    al, [vbe2bios] ; 0C0h-0C5h for emu (*)
 00003A11 3CC0          <1>       cmp    al, 0C0h ; BOCHS/QEMU/VIRTUALBOX (*) ?
 00003A13 7210          <1>       jb     short _vbe_rmi_3  ; cf = 1
 00003A15 3CC5          <1>       cmp    al, 0C5h ; (*)
 00003A17 770B          <1>       ja     short _vbe_rmi_2 ; unknown vbios !?
                        <1>
                        <1>       ;xor   edi, edi  ; 0 = kernel call (sign)
                        <1>       ;      ; no transfer to user's buffer
                        <1>
                        <1>       ;mov   ax, 4F01h
                        <1>
```

```
                                <1>       ; cx = Video mode (for 4F01h, with LFB flag)
                                <1>
00003A19 E80A000000             <1>       call   vbe_biosfn_return_mode_info
00003A1E 6683F84F               <1>       cmp    ax, 004Fh ; successful ?
00003A22 7401                   <1>       je     short _vbe_rmi_3  ; cf = 0
                                <1> _vbe_rmi_2:
00003A24 F9                     <1>       stc
                                <1>       ; cf = 1
                                <1> _vbe_rmi_3:
00003A25 5E                     <1>       pop    esi ; ****
00003A26 5D                     <1>       pop    ebp ; ***
                                <1>
                                <1>       ;pop   es  ; **
                                <1>       ;pop   ss  ; *
                                <1>
00003A27 C3                     <1>       retn
                                <1>
                                <1>
                                <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
                                <1> ; * ------------------------------------------------------
                                <1> ; * Function 01h - Return VBE Mode Information
                                <1> ; * ------------------------------------------------------
                                <1> ; * Input:
                                <1> ; *         AX = 4F01h
                                <1> ; *         CX = Mode number
                                <1> ; *    (ES:DI) EDI = Pointer to ModeInfoBlock structure
                                <1> ; * Output:
                                <1> ; *         AX = VBE Return Status
                                <1> ; *
                                <1> ; *------------------------------------------------------
                                <1> ; *
                                <1>
                                <1> vbe_biosfn_return_mode_info:
                                <1>       ; 15/12/2020
                                <1>       ; 14/12/2020
                                <1>       ; 12/12/2020
                                <1>       ; 11/12/2020 (TRDOS 386 v2.0.3)
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;      cx = video (bios) mode
                                <1>       ;      edi = ModeInfoBlock buffer address
                                <1>       ;           (in user's memory space)
                                <1>       ;      (ax = 4F01h)
                                <1>       ; Output:
                                <1>       ;      ax = 004Fh (successful)
                                <1>       ;      ah > 0 -> error
                                <1>       ;
                                <1>       ; Modified registers: esi
                                <1>
                                <1>       ;;push ds  ; *
                                <1>       ;;push es  ; **
                                <1>       ;;push ebp ; ***
                                <1>       ;;push esi ; ****
                                <1>
00003A28 F6C501                 <1>       test   ch, 1
00003A2B 7505                   <1>       jnz    short vbe_rmi_1
                                <1>
                                <1>       ; mode number < 100h
                                <1>       ; CGA/VGA mode is not proper this VBE function
                                <1>
00003A2D 29C0                   <1>       sub    eax, eax
                                <1> vbe_rmi_0:
                                <1>       ;mov   ax, 0100h  ; Function is not supported
00003A2F B401                   <1>       mov    ah, 1
00003A31 C3                     <1>       retn
                                <1> vbe_rmi_1:
00003A32 52                     <1>       push   edx ; *****
00003A33 51                     <1>       push   ecx ; ******
00003A34 53                     <1>       push   ebx ; *******
00003A35 57                     <1>       push   edi ; ********
                                <1>
                                <1>       ; 14/12/2020
00003A36 89CB                   <1>       mov    ebx, ecx
                                <1>
                                <1>       ;xor   eax, eax
00003A38 80E7C1                 <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
00003A3B 883D[1D120300]         <1>       mov    [vbe_mode_x], bh
                                <1>       ;and   bx, 1FFh
00003A41 80E701                 <1>       and    bh, 1
                                <1>       ;mov   bh, 1
                                <1>
                                <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
00003A44 E88A060000             <1>       call   set_mode_info_list ; (alternative 2)
                                <1>
                                <1>       ; eax = 0
                                <1>
                                <1>       ;mov   bx, [esi] ; mode
                                <1>
                                <1>       ; Alternative 1 (instead of 'set_mode_info_list')
                                <1>       ;call  mode_info_find_mode ; (alternative 1)
                                <1>
00003A49 09F6                   <1>       or     esi, esi
                                <1>       ; 14/12/2020
00003A4B 744D                   <1>       jz     short vbe_rmi_4     ; VBE mode number is wrong
                                <1>                                  ; or it is not supported
                                <1>
                                <1>       ; 15/12/2020
                                <1>       ;mov   bx, [esi] ; mode
                                <1>
                                <1>       ; 12/12/2020
                                <1>       ;call  set_lfbinfo_table
                                <1>
00003A4D F605[1D120300]40       <1>       test   byte [vbe_mode_x], 40h ; LFB model ?
00003A54 7404                   <1>       jz     short vbe_rmi_2
                                <1>
```

```
8241 00003A56 C6461C01            <1>       mov   byte [esi+MODEINFO.NumberOfBanks], 1
8242                              <1> vbe_rmi_2:
8243                              <1>       ; (vbe.c, 02/01/2020, vruppert)
8244                              <1>       ; 11/12/2020 (Erdogan Tan, video.s)
8245                              <1>       ; Bochs Graphics Adapter
8246                              <1>       ; vendor_id: 1111h, device id: 1234h
8247                              <1>
8248 00003A5A E855070000          <1>       call  pci_get_lfb_addr
8249                              <1>       ;or   eax, eax
8250 00003A5F 7404                <1>       jz    short vbe_rmi_3
8251                              <1>       ; zf = 0, ax > 0 (high word of LFB address)
8252                              <1>       ; set/change LFB address in MODEINFO structure
8253 00003A61 6689462C            <1>       mov   [esi+MODEINFO.PhysBasePtr+2], ax
8254                              <1>       ; 12/12/2020
8255                              <1>       ;mov  [edi+LFBINFO.LFB_addr+2], ax
8256                              <1> vbe_rmi_3:
8257                              <1>       ;test  byte [esi+MODEINFO.WinAAttributes], 1
8258                              <1>       ;            ; VBE_WINDOW_ATTRIBUTE_RELOCATABLE = 1
8259                              <1>         ;jz  short vbe_rmi_4
8260                              <1>       ;; 11/12/2020
8261                              <1>       ;; In fact, this is far call address in (Bochs/BGA) Video Bios
8262                              <1>       ;; Direct user access to kernel subroutines is not possible
8263                              <1>       ;; in TRDOS 386. Also, TRDOS 386 kernel will support only LFB.
8264                              <1>       ;; Bank select may be a seperate sysvideo function in future
8265                              <1>       ;; (if it will be required).
8266                              <1>       ;mov  dword [esi+MODEINFO.WinFuncPtr], dispi_set_bank_farcall
8267                              <1> ;vbe_rmi_4:
8268                              <1>       ; 12/12/2020
8269 00003A65 E83D000000          <1>       call  set_lfbinfo_table
8270                              <1>
8271                              <1>       ; 11/12/2020
8272                              <1>       ; copy 68 bytes of MODE_INFO_LIST to user
8273                              <1>
8274 00003A6A 8B3C24              <1>       mov   edi, [esp]  ; user's buffer address
8275                              <1>       ; 12/12/2020
8276 00003A6D 09FF                <1>       or    edi, edi ; 0 = kernel call
8277                              <1>                      ; (call from '_vbe_biosfn_return_mode_info')
8278 00003A6F 7432                <1>       jz    short vbe_rmi_6
8279                              <1>
8280                              <1>       ; 15/12/2020
8281                              <1>       ; prepare 256 bytes MODEINFO buffer at VBE3MODEINFOBLOCK
8282                              <1>       ; and then, copy buffer conttent to user's buffer
8283 00003A71 57                  <1>       push  edi
8284 00003A72 BE[3C120300]        <1>       mov   esi, MODE_INFO_LIST + 2   ; MODEINFO.ModeAttributes
8285 00003A77 BF007C0900          <1>       mov   edi, VBE3MODEINFOBLOCK
8286 00003A7C B910000000          <1>       mov   ecx, 66/4  ; 66 bytes
8287 00003A81 F3A5                <1>       rep   movsd
8288 00003A83 31C0                <1>       xor   eax, eax
8289 00003A85 B12F                <1>       mov   cl, (256-68)/4 ; 188 bytes
8290 00003A87 F3AB                <1>       rep   stosd
8291 00003A89 66AB                <1>       stosw ; 2 bytes
8292 00003A8B 5F                  <1>       pop   edi
8293 00003A8C BE007C0900          <1>       mov   esi, VBE3MODEINFOBLOCK
8294                              <1>       ;mov  cx, 256
8295 00003A91 FEC5                <1>       inc   ch ; cx = 256
8296 00003A93 E8AFE00000          <1>       call  transfer_to_user_buffer
8297 00003A98 7309                <1>       jnc   short vbe_rmi_5
8298                              <1> vbe_rmi_4:
8299                              <1>       ;mov  eax, 014Fh ; fail/error
8300 00003A9A 31C0                <1>       xor   eax, eax
8301 00003A9C B401                <1>       mov   ah, 01h
8302                              <1>       ;jmp  short vbe_rmi_6
8303 00003A9E E981000000          <1>       jmp   vbe_sm_ret1 ; 11/12/2020
8304                              <1> vbe_rmi_5:
8305                              <1>       ; 256 bytes of MODEINFO have been transferred to user
8306                              <1>       ;mov  eax, 4Fh ; succesfull
8307                              <1> vbe_rmi_6: ; 12/12/2020
8308 00003AA3 31C0                <1>       xor   eax, eax
8309                              <1> ;vbe_rmi_6:
8310 00003AA5 EB7D                <1>       jmp   vbe_sm_ret1 ; 11/12/2020
8311                              <1>
8312                              <1>       ;pop  edi ; ********
8313                              <1>       ;pop  ebx ; *******
8314                              <1>       ;pop  ecx ; ******
8315                              <1>       ;pop  edx ; *****
8316                              <1>
8317                              <1>       ;;pop  esi ; ****
8318                              <1>       ;;pop  ebp ; ***
8319                              <1>       ;;pop  es  ; **
8320                              <1>       ;;pop  ds  ; *
8321                              <1>
8322                              <1>       ;retn
8323                              <1>
8324                              <1> set_lfbinfo_table:
8325                              <1>       ; 19/12/2020
8326                              <1>       ; 11/12/2020
8327                              <1>       ; Set/Fill LFBINFO structure/table
8328                              <1>       ;
8329                              <1>       ; Input:
8330                              <1>       ;     esi = Mode info list address
8331                              <1>       ; Output:
8332                              <1>       ;     LFB_Info address is filled with LFBINFO
8333                              <1>       ;     edi = LFB_Info address
8334                              <1>       ;
8335                              <1>       ; Modified registers: eax, edx (=0), edi
8336                              <1>
8337 00003AA7 BF[2A120300]        <1>       mov   edi, LFB_Info
8338 00003AAC 8B462A              <1>       mov   eax, [esi+MODEINFO.PhysBasePtr]
8339 00003AAF 894702              <1>       mov   [edi+LFBINFO.LFB_addr], eax ; LFB address
8340                              <1>       ;mov  ax, [esi+MODEINFO.mode]
8341 00003AB2 668B06              <1>       mov   ax, [esi]
8342 00003AB5 668907              <1>       mov   [edi+LFBINFO.mode],ax
8343 00003AB8 8A461B              <1>       mov   al, [esi+MODEINFO.BitsPerPixel]
8344 00003ABB 88470E              <1>       mov   [edi+LFBINFO.bpp], al
8345 00003ABE 29C0                <1>       sub   eax, eax
```

```
8346 00003AC0 668B4614            <1>       mov    ax, [esi+MODEINFO.XResolution]
8347 00003AC4 6689470A            <1>       mov    [edi+LFBINFO.X_res], ax
8348 00003AC8 89C2                <1>       mov    edx, eax ; 19/12/2020
8349 00003ACA 668B4616            <1>       mov    ax, [esi+MODEINFO.YResolution]
8350 00003ACE 6689470C            <1>       mov    [edi+LFBINFO.Y_res], ax
8351                              <1>       ; eax = Y_res ; screen height
8352                              <1>       ; 19/12/2020
8353 00003AD2 F7E2                <1>       mul    edx ; X_res*Y_res
8354                              <1>       ; edx = 0
8355 00003AD4 8A570E              <1>       mov    dl, [edi+LFBINFO.bpp]
8356                              <1>       ; Note:
8357                              <1>       ; Bits per pixel may be 8,16,24,32 for TRDOS 386 v2.
8358                              <1>       ; (4 bits for pixel is not used for VESA modes here)
8359 00003AD7 C0EA03              <1>       shr    dl, 3 ; convert bits to byte
8360 00003ADA F7E2                <1>       mul    edx
8361                              <1>       ; eax = screen/page/buffer size in bytes
8362 00003ADC 894706              <1>       mov    [edi+LFBINFO.LFB_size], eax
8363                              <1>       ; edx = 0
8364                              <1>       ; clear reserved byte in LFBINFO structure/table
8365 00003ADF 88570F              <1>       mov    [edi+LFBINFO.reserved], dl ; not necessary
8366 00003AE2 C3                  <1>       retn
8367                              <1>
8368                              <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
8369                              <1> ; * ------------------------------------------------------
8370                              <1> ; * Function 02h - Set VBE Mode
8371                              <1> ; * ------------------------------------------------------
8372                              <1> ; * Input:
8373                              <1> ; *         AX = 4F02h
8374                              <1> ; *         BX = Desired Mode to set
8375                              <1> ; * Output:
8376                              <1> ; *         AX = VBE Return Status
8377                              <1> ; *
8378                              <1> ; *------------------------------------------------------
8379                              <1> ; *
8380                              <1>
8381                              <1> vbe_biosfn_set_mode:
8382                              <1>       ; 12/12/2020
8383                              <1>       ; 11/12/2020 (LFBINFO table for VESA VBE modes)
8384                              <1>       ; 27/11/2020
8385                              <1>       ; 25/11/2020
8386                              <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
8387                              <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
8388                              <1>       ;
8389                              <1>       ; Input:
8390                              <1>       ;      bx = video (bios) mode
8391                              <1>       ;      ax = 4F02h
8392                              <1>       ; Output:
8393                              <1>       ;      ax = 004Fh (successful)
8394                              <1>       ;      ah > 0 -> error
8395                              <1>       ;
8396                              <1>       ; Modified registers: esi
8397                              <1>
8398                              <1>       ; 27/11/2020
8399                              <1>
8400                              <1>       ;;push ds  ; *
8401                              <1>       ;;push es  ; **
8402                              <1>       ;;push ebp ; ***
8403                              <1>       ;;push esi ; ****
8404                              <1>
8405                              <1>       ; 11/12/2020
8406 00003AE3 52                  <1>       push   edx ; *****
8407 00003AE4 51                  <1>       push   ecx ; ******
8408 00003AE5 53                  <1>       push   ebx ; *******
8409 00003AE6 57                  <1>       push   edi ; ********
8410                              <1>
8411                              <1>       ;xor   eax, eax
8412 00003AE7 80E7C1              <1>       and    bh, 0C1h ; use bit 15, 14, 8 only (for bh)
8413 00003AEA 883D[1D120300]      <1>       mov    [vbe_mode_x], bh
8414 00003AF0 80E701              <1>       and    bh, 1
8415 00003AF3 753C                <1>       jnz    short vbe_sm_3  ; VESA VBE mode
8416                              <1>
8417                              <1>       ;;test bx, 4000h ; VBE_MODE_LINEAR_FRAME_BUFFER
8418                              <1>       ;test  bh, 40h
8419                              <1>       ;jz    short vbe_sm_0
8420                              <1>       ;; lfb_flag
8421                              <1>       ;mov   al, 40h      ; VBE_DISPI_LFB_ENABLED
8422                              <1> vbe_sm_0:
8423                              <1>       ; 27/11/2020
8424 00003AF5 B080                <1>       mov    al, 80h
8425                              <1>       ;test  bh, 80h ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
8426                              <1>       ;jnz   short vbe_sm_1 ; no_clear
8427                              <1>       ;; clear
8428                              <1>       ;sub   al, al ; 0
8429 00003AF7 8405[1D120300]      <1>       test   [vbe_mode_x], al ; 80h
8430 00003AFD 7402                <1>       jz     short vbe_sm_1 ; clear display memory
8431                              <1>       ; no_clear
8432 00003AFF 08C3                <1>       or     bl, al ; VBE_MODE_PRESERVE_DISPLAY_MEMORY
8433                              <1> vbe_sm_1:
8434                              <1>       ; check non vesa mode
8435                              <1>       ;;cmp  bx, 100h ; VBE_MODE_VESA_DEFINED
8436                              <1>       ;;jna  short vbe_sm_2
8437                              <1>       ;and   bh, 1
8438                              <1>       ;jnz   short vbe_sm_3
8439                              <1>
8440                              <1>       ; BX <= 1FFh
8441                              <1>
8442                              <1>       ; 27/11/2020
8443                              <1>       ;or    bl, al ; al = 80h if no_clear option is set
8444                              <1>       ;             ; al = 0 if no_clear option is not set
8445                              <1>
8446                              <1>       ; 25/11/2020
8447                              <1>       ; VBE DISPI will be disabled in 'biosfn_set_video_mode'
8448                              <1>
8449                              <1>       ;xor   al, al ; 0 ; VBE_DISPI_DISABLED
8450                              <1>       ;call  dispi_set_enable
```

```
                                <1>
                                <1>       ; call the vgabios in order to set the video mode
                                <1>       ; this allows for going back to textmode with a VBE call
                                <1>       ; (some applications expect that to work)
                                <1>
                                <1>       ;and   bx, 0FFh
                                <1>
                                <1>       ; 27/11/2020
                                <1> biosfn_set_video_mode:
                                <1>       ; _call: call subroutine
                                <1>       ; 26/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; (ref: vgabios.c, 02/01/2020, vruppert)
                                <1>       ; Input:
                                <1>       ;      bl = VGA video (bios) mode
                                <1>       ; Output:
                                <1>       ;      cf = 1 -> error
                                <1>       ;      cf = 0 -> ok
                                <1>       ;
                                <1>       ; Modified registers: esi
                                <1>
                                <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
                                <1>
                                <1>       ;mov   ax, 0 ; VBE_DISPI_DISABLED
00003B01 31C0                   <1>       xor    eax, eax ; 0
00003B03 E8A3040000             <1>       call   dispi_set_enable
                                <1>
00003B08 88D8                   <1>       mov    al, bl
                                <1>       ;jmp   _set_mode ; (in 'biosfn_set_video_mode' sub)
00003B0A E861E0FFFF             <1>       call   _set_mode ; will return with cf=1 only if
                                <1>                        ; desired mode is not implemented
                                <1>       ; _retn: return from subroutine
00003B0F 721A                   <1>       jc     short vbe_sm_2 ; 25/11/2020
                                <1>
                                <1>       ; 26/11/2020
00003B11 31C0                   <1>       xor    eax, eax
00003B13 A0[DA6F0000]           <1>       mov    al, [CRT_MODE]
                                <1>       ; 27/11/2020
00003B18 8A25[AF960100]         <1>       mov    ah, [noclearmem] ; 80h or 0
                                <1>       ;and   ah 80h
00003B1E 66A3[1E120300]         <1>       mov    [video_mode], ax ; bit 15 = no_clear flag
                                <1>                               ; bit 14 = 0 (not LFB model)
                                <1> vbe_sm_ret1:
                                <1>       ; 11/12/2020
                                <1>       ; (vbe_rmi_4 and vbe_rmi_6 jump here)
                                <1>       ; 27/11/2020
00003B24 B04F                   <1>       mov    al, 4Fh ; Function call successful
                                <1>       ; eax = 004Fh
                                <1> vbe_sm_ret2:
                                <1>       ; 11/12/2020
00003B26 5F                     <1>       pop    edi ; ********
00003B27 5B                     <1>       pop    ebx ; *******
00003B28 59                     <1>       pop    ecx ; ******
00003B29 5A                     <1>       pop    edx ; *****
                                <1>
                                <1>       ;;pop  esi ; ****
                                <1>       ;;pop  ebp ; ***
                                <1>       ;;pop  es  ; **
                                <1>       ;;pop  ds  ; *
                                <1>
00003B2A C3                     <1>       retn
                                <1>
                                <1> vbe_sm_2:
                                <1>       ;mov   ax, 0100h ; Function is not supported
                                <1>       ; 27/11/2020
00003B2B 31C0                   <1>       xor    eax, eax
00003B2D B401                   <1>       mov    ah, 01h
                                <1>       ; eax = 0100h
                                <1>       ;retn
00003B2F EBF5                   <1>       jmp    short vbe_sm_ret2
                                <1>
                                <1> vbe_sm_3:
                                <1>       ; 12/12/2020
                                <1>       ; check current mode, if it is 03h
                                <1>       ; save page contents and cursor positions
00003B31 803D[DA6F0000]03       <1>       cmp    byte [CRT_MODE], 03h
00003B38 75BB                   <1>       jne    short vbe_sm_0
00003B3A E8E6E2FFFF             <1>       call   save_mode3_multiscreen
                                <1>       ; set current mode to extended (SVGA) mode
                                <1>       ;mov   byte [CRT_MODE], 0FFh ; VESA VBE mode
                                <1> vbe_sm_4:
                                <1>       ; 27/11/2020
                                <1>       ; bx = mode (bit 0 to 8)
                                <1>
                                <1>       ; 25/11/2020
                                <1>
                                <1>       ; Alternative 2 (instead of 'Mode_info_find_mode')
                                <1>       ;push  edi
00003B3F E88F050000             <1>       call   set_mode_info_list ; (alternative 2)
                                <1>       ;pop   edi
                                <1>
                                <1>       ;mov   bx, [esi] ; mode
                                <1>
                                <1>       ; Alternative 1 (instead of 'set_mode_info_list')
                                <1>       ;call  mode_info_find_mode ; (alternative 1)
                                <1>
00003B44 09F6                   <1>       or     esi, esi
00003B46 74E3                   <1>       jz     short vbe_sm_2 ; VBE mode number is wrong
                                <1>                             ; or it is not supported
                                <1>
                                <1>       ; 11/12/2020
00003B48 668B1E                 <1>       mov    bx, [esi] ; mode
                                <1>
                                <1>       ; 27/11/2020
00003B4B 0A3D[1D120300]         <1>       or     bh, [vbe_mode_x]
                                <1>
```

```
8556                                    <1>       ; save VESA VBE mode
8557 00003B51 66891D[1E120300]          <1>       mov    [video_mode], bx
8558                                    <1>              ; 27/11/2020
8559                                    <1>              ; bit 0 to 8 = VESA VBE mode
8560                                    <1>              ; bit 9 to 13 = 0 (bit 0 to 13 = mode)
8561                                    <1>              ; bit 14 = Linear/Flat Frame Buffer flag
8562                                    <1>              ; bit 15 = 'memory not cleared
8563                                    <1>              ;         at last mode set' flag
8564                                    <1>
8565                                    <1>       ; first disable current mode
8566                                    <1>       ; (when switching between vesa modes)
8567                                    <1>       ; 'dispi_set_enable(VBE_DISPI_DISABLED);'
8568                                    <1>
8569                                    <1>       ;mov   ax, VBE_DISPI_DISABLED ; 0
8570 00003B58 29C0                     <1>       sub    eax, eax ; 0
8571                                    <1>
8572 00003B5A E84C040000               <1>       call   dispi_set_enable
8573                                    <1>
8574                                    <1>       ; 11/12/2020
8575 00003B5F 8A461B                   <1>       mov    al, [esi+MODEINFO.BitsPerPixel]
8576                                    <1>       ; ah = 0
8577                                    <1>
8578                                    <1>       ;cmp   byte [esi+MODEINFO.BitsPerPixel], 8
8579 00003B62 3C08                     <1>       cmp    al, 8
8580 00003B64 750B                     <1>       jne    short vbe_sm_5
8581                                    <1>
8582                                    <1>       ; 11/12/2020
8583                                    <1>       ;push  edi
8584 00003B66 50                       <1>       push   eax
8585                                    <1>       ; 'load_dac_palette(3);'
8586 00003B67 56                       <1>       push   esi
8587 00003B68 B403                     <1>       mov    ah, 3  ; palette3, 256 colors
8588 00003B6A E828F1FFFF               <1>       call   load_dac_palette
8589 00003B6F 5E                       <1>       pop    esi
8590                                    <1>       ; 11/12/2020
8591 00003B70 58                       <1>       pop    eax
8592                                    <1>       ;pop   edi
8593                                    <1> vbe_sm_5:
8594                                    <1>       ;'dispi_set_bpp(cur_info->info.BitsPerPixel);'
8595                                    <1>       ; 11/12/2020 (al = bits per pixel, ah = 0)
8596                                    <1>       ;xor   ah, ah
8597                                    <1>       ;mov   al, [esi+MODEINFO.BitsPerPixel]
8598 00003B71 E849040000               <1>       call   dispi_set_bpp
8599                                    <1>         ;'dispi_set_xres(cur_info->info.XResolution);'
8600 00003B76 668B4614                 <1>       mov    ax, [esi+MODEINFO.XResolution]
8601 00003B7A E846040000               <1>       call   dispi_set_xres
8602                                    <1>         ;'dispi_set_yres(cur_info->info.YResolution);'
8603 00003B7F 668B4616                 <1>       mov    ax, [esi+MODEINFO.YResolution]
8604 00003B83 E843040000               <1>       call   dispi_set_yres
8605                                    <1>
8606                                    <1>       ;'dispi_set_bank(0);'
8607                                    <1>       ;xor   ax, ax
8608 00003B88 31C0                     <1>       xor    eax, eax ; 0
8609 00003B8A E842040000               <1>       call   dispi_set_bank
8610                                    <1>         ;'dispi_set_enable(VBE_DISPI_ENABLED|no_clear|lfb_flag);'
8611                                    <1>       ;mov   ax, di
8612                                    <1>       ; ah = 0 ; 27/11/2020
8613 00003B8F A0[1D120300]             <1>       mov    al, [vbe_mode_x] ; restore VBE mode bit 14 & 15
8614 00003B94 0C01                     <1>       or     al, 1 ; VBE_DISPI_ENABLED
8615 00003B96 E810040000               <1>       call   dispi_set_enable
8616                                    <1>
8617                                    <1>         ; 'vga_compat_setup();'
8618 00003B9B E846040000               <1>       call   vga_compat_setup
8619                                    <1>
8620                                    <1>       ; 11/12/2020
8621 00003BA0 E802FFFFFF               <1>       call   set_lfbinfo_table
8622                                    <1>
8623                                    <1>       ; 26/11/2020
8624 00003BA5 31C0                     <1>       xor    eax, eax
8625 00003BA7 FEC8                     <1>       dec    al
8626 00003BA9 A2[DA6F0000]             <1>       mov    [CRT_MODE], al ; 0FFh = VESA VBE mode sign
8627                                    <1>
8628                                    <1>       ; 27/11/2020
8629 00003BAE E971FFFFFF               <1>       jmp    vbe_sm_ret1 ; Function call successful
8630                                    <1>
8631                                    <1>       ; 27/11/2020
8632                                    <1>       ;mov   al, 4Fh
8633                                    <1>       ;      ; eax = 004Fh = Function call successful
8634                                    <1>       ;jmp   short vbe_sm_ret2
8635                                    <1>
8636                                    <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
8637                                    <1> ; * ------------------------------------------------------
8638                                    <1> ; * Function 03h - Return Current VBE Mode
8639                                    <1> ; * ------------------------------------------------------
8640                                    <1> ; * Input:
8641                                    <1> ; *          AX = 4F03h
8642                                    <1> ; * Output:
8643                                    <1> ; *          AX = VBE Return Status
8644                                    <1> ; *          BX = Current VBE Mode
8645                                    <1> ; *
8646                                    <1> ; *------------------------------------------------------
8647                                    <1> ; *
8648                                    <1>
8649                                    <1> vbe_biosfn_return_current_mode:
8650                                    <1>       ; 11/12/2020
8651                                    <1>       ; 27/11/2020 (TRDOS 386 v2.0.3)
8652                                    <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
8653                                    <1>       ;
8654                                    <1>       ; Input:
8655                                    <1>       ;      none
8656                                    <1>       ; Output:
8657                                    <1>       ;      ax = 004Fh (successful)
8658                                    <1>       ;      ah > 0 -> error
8659                                    <1>       ;      bx = current video (bios) mode (if ah = 0)
8660                                    <1>       ;
```

```
                                <1>       ; Modified registers: eax, ebx
                                <1>
                                <1>       ; 27/11/2020
                                <1>
                                <1>       ;;push ds  ; *
                                <1>       ;;push es  ; **
                                <1>       ;;push ebp ; ***
                                <1>       ;;push esi ; ****
                                <1>
                                <1>       ;push  edx ; *****
                                <1>
                                <1>       ; (vbe.c)
                                <1>       ;call  dispi_get_enable
                                <1>       ;      ; ax = vbe display interface status
                                <1>       ;and   al, 1 ; VBE_DISPI_ENABLED
                                <1>       ;jnz   short vbe_gm_1  ; VBE graphics mode
                                <1>
00003BB3 A0[DA6F0000]           <1>       mov    al, [CRT_MODE] ; current cga/vga mode
00003BB8 3CFF                   <1>       cmp    al, 0FFh ; VBE extension signature
00003BBA 720E                   <1>       jb     short vbe_gm_1 ; get CGA/VGA mode
                                <1>
                                <1>       ; get VBE mode
                                <1> vbe_gm_0:
00003BBC 66A1[1E120300]         <1>       mov    ax, [video_mode]
                                <1>              ; BX bits:
                                <1>              ; bit 0 to 8 = VESA VBE video mode
                                <1>              ; bit 9 to 13 = 0
                                <1>              ; bit 14 = last mode set LFB option
                                <1>              ;        1 - linear/flat frame buffer
                                <1>              ;        0 - windowed frame buffer
                                <1>              ; bit 15 = last mode set no_clear option
                                <1>              ;        0 - video memory cleared
                                <1>              ;        1 - video memory not cleared
                                <1>
                                <1> vbe_gm_return:
                                <1>       ;pop   edx ; ******
00003BC2 0FB7D8                 <1>       movzx  ebx, ax
                                <1> ;vbe_srs_retn:
00003BC5 31C0                   <1>       xor    eax, eax ; 0
00003BC7 B04F                   <1>       mov    al, 4Fh ; ax = 004Fh (successful)
00003BC9 C3                     <1>       retn
                                <1>
                                <1> vbe_gm_1:
                                <1>       ; legacy (old, standard) CGA/VGA bios video mode
00003BCA 8A25[AF960100]         <1>       mov    ah, [noclearmem] ; 80h or 0
                                <1>              ; BX bits:
                                <1>              ; bit 0 to 7 = video mode
                                <1>              ; bit 8 to 13 = 0
                                <1>              ; bit 14 = 0 (not LFB mode) CGA/VGA
                                <1>              ; bit 15 = 1 if [noclearmem] = 80h
                                <1>              ;         0 if [noclearmem] = 0
00003BD0 EBF0                   <1>       jmp    short vbe_gm_return
                                <1>
                                <1> ; * (TRDOS 386, INT 31h, VESA Video Bios functions)
                                <1> ; * ------------------------------------------------------
                                <1> ; * Function 04h - Save/Restore State
                                <1> ; * ------------------------------------------------------
                                <1> ; * Input:
                                <1> ; *          AX = 4F04h
                                <1> ; *             DL = 00h Return Save/Restore State buff size
                                <1> ; *                  01h Save State
                                <1> ; *                  02h Restore State
                                <1> ; *             CX = Requested states
                                <1> ; *               bit 0 - controller hardware state
                                <1> ; *               bit 1 - BIOS data state
                                <1> ; *               bit 2 - DAC state
                                <1> ; *               bit 3 - register state
                                <1> ; *    (ES:BX) EBX = Pointer to buffer (if DL <> 00h)
                                <1> ; * Output:
                                <1> ; *          AX = VBE Return Status
                                <1> ; *          BX = Number of 64-byte blocks
                                <1> ; *               to hold the state buffer (if DL=00h)
                                <1> ; *
                                <1> ; *------------------------------------------------------
                                <1> ; *
                                <1>
                                <1> vbe_biosfn_save_restore_state:
                                <1>       ; 23/01/2021
                                <1>       ; 16/01/2021
                                <1>       ; 14/01/2021
                                <1>       ; 13/01/2021
                                <1>       ; 12/01/2021
                                <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; (ref: vbe.c, 02/01/2020, vruppert)
                                <1>       ;
                                <1>       ; Input:
                                <1>       ;      dl = sub function
                                <1>       ;      cl = requested state
                                <1>       ;       ebx = pointer to buffer (if dl<>00h)
                                <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, edi
                                <1>
                                <1>       ; 14/01/2021
00003BD2 09DB                   <1>       or     ebx, ebx ; user's buffer address
00003BD4 750A                   <1>       jnz    short _vbe_biosfn_save_restore_state
                                <1>
00003BD6 20D2                   <1>       and    dl, dl
00003BD8 7406                   <1>       jz     short _vbe_biosfn_save_restore_state
                                <1>
                                <1>       ; function failed
```

```
                        <1>       ;mov   eax, 0100h
                        <1>       ;xor   eax, eax
                        <1>       ;inc   ah  ; eax = 0100h
                        <1>       ; 16/01/2021
00003BDA B84F010000     <1>       mov    eax, 014Fh
00003BDF C3             <1>       retn
                        <1>
                        <1> _vbe_biosfn_save_restore_state:
                        <1>       ; 23/01/2021
                        <1>       ; 14/01/2021
                        <1>       ; ebx = 0 if the caller is kernel ('sysvideo')
                        <1>
                        <1>       ; 13/01/2021
00003BE0 57             <1>       push   edi
00003BE1 52             <1>       push   edx
00003BE2 51             <1>       push   ecx
                        <1>
                        <1>       ; 23/01/2021
                        <1>       ; 12/01/2021
00003BE3 80FA02         <1>       cmp    dl, 2
00003BE6 7757           <1>       ja     short vbe_srs_7 ; 23/01/2021
                        <1>                      ; invalid sub function
00003BE8 83F90F         <1>       cmp    ecx, 0Fh
00003BEB 7752           <1>       ja     short vbe_srs_7 ; invalid !
                        <1>
00003BED 20D2           <1>       and    dl, dl
00003BEF 7515           <1>       jnz    short vbe_srs_4
                        <1>
                        <1>       ; DL = 0
                        <1>       ; Return Save/Restore State buffer size
                        <1>
                        <1>       ;mov   ebx, ecx
                        <1>       ;shl   bl, 1
                        <1>       ;mov   bx, [ebx+vbestatebufsize]
00003BF1 E881000000     <1>       call   vbe_srs_gbs
                        <1>
                        <1> ;     ; 11/01/2021
                        <1> ;     test   cl, 8
                        <1> ;     jz     short vbe_srs_3
                        <1> ;     ; vbe_biosfn_read_video_state_size();
                        <1> ;     ; return 9 * 2;
                        <1> ;     mov    bl, 18 ; register state size
                        <1> ;vbe_srs_0:
                        <1> ;     test   cl, 1
                        <1> ;     jz     short vbe_srs_1
                        <1> ;     ; size += 0x46;
                        <1> ;     add    bl, 70 ; controller state size
                        <1> ;vbe_srs_1:
                        <1> ;     test   cl, 2
                        <1> ;     jz     short vbe_srs_2
                        <1> ;     ; size += (5 + 8 + 5) * 2 + 6;
                        <1> ;     ;add   bl, 42 ; BIOS data state size ; Bochs/Plex86
                        <1> ;     ; 12/01/2021
                        <1> ;     add    bl, 40 ; TRDOS 386 v2 VBIOS data state size
                        <1> ;vbe_srs_2:
                        <1> ;     test   cl, 4
                        <1> ;     jz     short vbe_srs_3
                        <1> ;     ; size += 3 + 256 * 3 + 1;
                        <1> ;     add    bx, 772 ; DAC state size
                        <1>
                        <1> vbe_srs_3:
00003BF6 6683C33F       <1>       add    bx, 63
00003BFA 66C1EB06       <1>       shr    bx, 6 ; / 64
                        <1>
                        <1> vbe_srs_retn:
00003BFE 31C0           <1>       xor    eax, eax ; 0
                        <1> vbe_srs_0:  ; 16/01/2021
00003C00 B04F           <1>       mov    al, 4Fh ; ax = 004Fh (successful)
                        <1> ;vbe_srs_0:
                        <1>       ; 13/01/2021
00003C02 59             <1>       pop    ecx
00003C03 5A             <1>       pop    edx
00003C04 5F             <1>       pop    edi
                        <1>
00003C05 C3             <1>       retn
                        <1>
                        <1>       ; 23/01/2021
                        <1> ;vbe_srs_10:
                        <1>       ;; 14/01/2021
                        <1>       ; return to 'sysvideo'
                        <1>       ;mov   ebx, ecx ; transfer count
                        <1>       ;      ; (byte count for saving current video state)
                        <1>       ;jmp   short vbe_srs_retn
                        <1>
                        <1> vbe_srs_4:
                        <1>       ; 23/01/2021
00003C06 80E10F         <1>       and    cl, 0Fh ; 8, 4, 2, 1
00003C09 7434           <1>       jz     short vbe_srs_7 ; cx = 0 -> invalid !
                        <1>
00003C0B BF00760900     <1>       mov    edi, VBE3SAVERESTOREBLOCK
                        <1>
00003C10 80FA01         <1>       cmp    dl, 1
00003C13 7730           <1>       ja     short vbe_srs_8
                        <1>
                        <1>       ; save video state
                        <1>
00003C15 F6C107         <1>       test   cl, 07h ; 4, 2, 1
00003C18 740A           <1>       jz     short vbe_srs_5  ; vbe dispi regs state
                        <1>
00003C1A E884000000     <1>       call   biosfn_save_video_state
                        <1>       ; edi = current position
                        <1>       ;      in VBE3SAVERESTOREBLOCK
                        <1>       ;      (VGA save_state offset)
                        <1>       ; modified regs: edi, eax, edx, ch
00003C1F F6C108         <1>       test   cl, 8
```

```
8871 00003C22 7405              <1>       jz      short vbe_srs_6
8872                            <1> vbe_srs_5:
8873 00003C24 E8AC010000        <1>       call    vbe_biosfn_save_video_state
8874                            <1>       ; edi = end position
8875                            <1>       ;       in VBE3SAVERESTOREBLOCK
8876                            <1>       ;       (VGA save_state offset)
8877                            <1>       ; modified regs: edi, eax, edx, ch
8878                            <1> vbe_srs_6:
8879                            <1>       ; 23/01/2021
8880 00003C29 21DB              <1>       and     ebx, ebx
8881 00003C2B 74D1              <1>       jz      short vbe_srs_retn ; the caller is kernel
8882                            <1>
8883 00003C2D BE00760900        <1>       mov     esi, VBE3SAVERESTOREBLOCK
8884 00003C32 29F7              <1>       sub     edi, esi
8885 00003C34 89F9              <1>       mov     ecx, edi ; transfer count in bytes
8886                            <1>
8887                            <1>       ;; 14/01/2021
8888                            <1>       ;and    ebx, ebx
8889                            <1>       ;jz     short vbe_srs_10 ; the caller is kernel
8890                            <1>
8891 00003C36 89DF              <1>       mov     edi, ebx ; user's buffer address
8892 00003C38 E80ADF0000        <1>       call    transfer_to_user_buffer
8893 00003C3D 73BF              <1>       jnc     short vbe_srs_retn
8894                            <1> vbe_srs_7:
8895                            <1>       ; // function failed
8896                            <1>       ;mov    eax, 0100h
8897 00003C3F 31C0              <1>       xor     eax, eax
8898 00003C41 FEC4              <1>       inc     ah  ; eax = 0100h
8899                            <1>       ; 16/01/2021
8900                            <1>       ; ax = 0014Fh
8901                            <1>       ;retn
8902                            <1>       ; 13/01/2021
8903 00003C43 EBBB              <1>       jmp     short vbe_srs_0
8904                            <1> vbe_srs_8:
8905                            <1>       ;cmp    dl, 2
8906                            <1>       ;jne    short vbe_srs_7
8907                            <1>       ;             ; invalid sub function
8908                            <1>
8909                            <1>       ; 14/01/2021
8910 00003C45 09DB              <1>       or      ebx, ebx ; user's buffer address
8911                            <1>       ;jnz    short vbe_srs_11
8912                            <1>
8913                            <1>       ; the caller is kernel ('sysvideo')
8914                            <1>       ;jmp    short vbe_srs_12
8915                            <1>       ; 23/01/2021
8916 00003C47 7414              <1>       jz      short vbe_srs_12 ; 'sysvideo' call
8917                            <1> vbe_srs_11:
8918 00003C49 89DE              <1>       mov     esi, ebx ; user's buffer address
8919                            <1>       ; 23/01/2021
8920                            <1>       ;push   ebx
8921                            <1>
8922 00003C4B E827000000        <1>       call    vbe_srs_gbs
8923                            <1>
8924                            <1>       ; restore video state
8925                            <1>
8926                            <1>       ;mov    edi, VBE3SAVERESTOREBLOCK
8927 00003C50 51                <1>       push    ecx
8928 00003C51 89D9              <1>       mov     ecx, ebx ; transfer count in bytes
8929 00003C53 E839DF0000        <1>       call    transfer_from_user_buffer
8930 00003C58 59                <1>       pop     ecx
8931                            <1>       ; 23/01/2021
8932                            <1>       ;pop    ebx
8933 00003C59 89F3              <1>       mov     ebx, esi
8934 00003C5B 72E2              <1>       jc      short vbe_srs_7
8935                            <1>
8936                            <1> vbe_srs_12:
8937                            <1>       ;mov    esi, VBE3SAVERESTOREBLOCK
8938 00003C5D 89FE              <1>       mov     esi, edi
8939                            <1>
8940 00003C5F F6C107            <1>       test    cl, 07h ; 4, 2, 1
8941 00003C62 740C              <1>       jz      short vbe_srs_9     ; vbe dispi regs state
8942                            <1>
8943 00003C64 E8A8010000        <1>       call    biosfn_restore_video_state
8944 00003C69 72D4              <1>       jc      short vbe_srs_7 ; invalid buffer content !
8945                            <1>       ; esi = current position
8946                            <1>       ;       in VBE3SAVERESTOREBLOCK
8947                            <1>       ;       (VGA save_state offset)
8948                            <1>       ; modified regs: esi, eax, edx, ch
8949 00003C6B F6C108            <1>       test    cl, 8
8950                            <1>       ;jz     short vbe_srs_10
8951                            <1>       ; 23/01/2020
8952 00003C6E EB8E              <1>       jmp     short vbe_srs_retn
8953                            <1> vbe_srs_9:
8954 00003C70 E8F8020000        <1>       call    vbe_biosfn_restore_video_state
8955                            <1>
8956                            <1>       ; modified regs: esi, eax, edx, ch
8957                            <1>
8958 00003C75 EB87              <1>       jmp     short vbe_srs_retn
8959                            <1>
8960                            <1> ;vbe_srs_10:
8961                            <1> ;     ; successful
8962                            <1> ;     xor     eax, eax ; 0
8963                            <1> ;     mov     al, 4Fh ; ax = 004Fh (successful)
8964                            <1> ;     retn
8965                            <1>
8966                            <1> vbe_srs_gbs:
8967                            <1>       ; return buffer size according to flags
8968 00003C77 89CB              <1>       mov     ebx, ecx ; options/flags
8969 00003C79 D0E3              <1>       shl     bl, 1
8970 00003C7B 668B9B[833C0000]  <1>       mov     bx, [ebx+vbestatebufsize]
8971 00003C82 C3                <1>       retn
8972                            <1>
8973                            <1> vbestatebufsize:
8974                            <1>       ; ----------------------------------------
8975                            <1>       ; CL = 0  1   2    3    4    5    6    7
```

```
                                <1>       ; ----------------------------------------
00003C83 0000460028006E0004-    <1>       dw     0, 70, 40, 110, 772, 842, 812, 882
00003C8C 034A032C037203         <1>
                                <1>       ; ----------------------------------------
                                <1>       ; CL = 8    9   10  11   12   13   14   15
                                <1>       ; ----------------------------------------
00003C93 120058003A00800016-    <1>       dw     18, 88, 58, 128, 790, 860, 830, 900
00003C9C 035C033E038403         <1>
                                <1>
                                <1> ; 11/01/2021
                                <1> VGAREG_ACTL_ADDRESS       equ 3C0h
                                <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
                                <1> VGAREG_ACTL_READ_DATA     equ 3C1h
                                <1>
                                <1> VGAREG_INPUT_STATUS       equ 3C2h
                                <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
                                <1> VGAREG_VIDEO_ENABLE       equ 3C3h
                                <1> VGAREG_SEQU_ADDRESS       equ 3C4h
                                <1> VGAREG_SEQU_DATA   equ 3C5h
                                <1>
                                <1> VGAREG_PEL_MASK           equ 3C6h
                                <1> VGAREG_DAC_STATE   equ 3C7h
                                <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
                                <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
                                <1> VGAREG_DAC_DATA           equ 3C9h
                                <1>
                                <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
                                <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
                                <1>
                                <1> VGAREG_GRDC_ADDRESS       equ 3CEh
                                <1> VGAREG_GRDC_DATA   equ 3CFh
                                <1>
                                <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
                                <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
                                <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
                                <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
                                <1>
                                <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
                                <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
                                <1> VGAREG_ACTL_RESET  equ 3DAh
                                <1>
                                <1> ;VGAREG_MDA_MODECTL       equ 3B8h
                                <1> VGAREG_CGA_MODECTL equ 3D8h
                                <1> VGAREG_CGA_PALETTE equ 3D9h
                                <1>
                                <1> biosfn_save_video_state:
                                <1>       ; 22/01/2021
                                <1>       ; 12/01/2021
                                <1>       ; 11/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; (vgabios.c)
                                <1>
                                <1>       ; modified registers: eax, edx, edi, ch
                                <1>
                                <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
                                <1>
                                <1>       ; input: edi = state buffer address
                                <1>
00003CA3 F6C101                 <1>       test   cl, 1
00003CA6 0F8485000000           <1>       jz     bfn_svs_4
                                <1>
00003CAC 66BAC403               <1>       mov    dx, VGAREG_SEQU_ADDRESS ; 3C7h
00003CB0 EC                     <1>       in     al, dx
00003CB1 AA                     <1>       stosb
                                <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
00003CB2 B2D4                   <1>       mov    dl, 0D4h
00003CB4 EC                     <1>       in     al, dx
00003CB5 AA                     <1>       stosb
                                <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
00003CB6 B2CE                   <1>         mov  dl, 0CEh
00003CB8 EC                     <1>       in     al, dx
00003CB9 AA                     <1>       stosb
                                <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
00003CBA B2DA                   <1>       mov    dl, 0DAh
00003CBC EC                     <1>       in     al, dx
                                <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
00003CBD B2C0                   <1>       mov    dl, 0C0h
00003CBF EC                     <1>       in     al, dx
00003CC0 AA                     <1>       stosb
00003CC1 88C4                   <1>       mov    ah, al ; ar_index
                                <1>       ;mov   dx, VGAREG_READ_FEATURE_CTL ; 3CAh
00003CC3 B2CA                   <1>       mov    dl, 0CAh
00003CC5 EC                     <1>       in     al, dx
00003CC6 AA                     <1>       stosb
                                <1>       ; (5 bytes are writen above)
                                <1>
                                <1>       ; for(i=1;i<=4;i++){
00003CC7 B001                   <1>       mov    al, 1
                                <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
                                <1>       ;mov   dl, 0C4h
00003CC9 B504                   <1>       mov    ch, 4
                                <1> bfn_svs_0:
                                <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
                                <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
00003CCB B2C4                   <1>       mov    dl, 0C4h
00003CCD EE                     <1>       out    dx, al
                                <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
00003CCE FEC2                   <1>       inc    dl  ; dx = 3C5h
                                <1>       ; inb(VGAREG_SEQU_DATA)
00003CD0 50                     <1>       push   eax
00003CD1 EC                     <1>       in     al, dx
00003CD2 AA                     <1>       stosb  ; (4 bytes in loop)
00003CD3 58                     <1>       pop    eax
                                <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
                                <1>       ;dec   dl
00003CD4 FEC0                   <1>       inc    al  ; i++
```

```
9079 00003CD6 FECD              <1>       dec    ch
9080 00003CD8 75F1              <1>       jnz    short bfn_svs_0
9081                            <1>
9082                            <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
9083 00003CDA 28C0              <1>       sub    al, al ; 0
9084 00003CDC EE                <1>       out    dx, al
9085                            <1>       ; inb(VGAREG_SEQU_DATA)
9086                            <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
9087 00003CDD FEC2              <1>       inc    dl  ; dx = 3C5h
9088 00003CDF EC                <1>       in     al, dx
9089 00003CE0 AA                <1>       stosb  ; (+1 byte)
9090                            <1>
9091                            <1>         ; for(i=0;i<=0x18;i++) {
9092 00003CE1 28C0              <1>       sub    al, al ; 0
9093                            <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9094                            <1>       ;mov   dl, 0D4h
9095 00003CE3 B519              <1>       mov    ch, 25
9096                            <1> bfn_svs_1:
9097                            <1>         ; outb(crtc_addr,i);
9098                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9099 00003CE5 B2D4              <1>       mov    dl, 0D4h
9100 00003CE7 EE                <1>       out    dx, al
9101                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
9102 00003CE8 FEC2              <1>       inc    dl  ; dx = 3D5h
9103                            <1>       ; inb(crtc_addr+1)
9104 00003CEA 50                <1>       push   eax
9105 00003CEB EC                <1>       in     al, dx
9106 00003CEC AA                <1>       stosb  ; (25 bytes in loop)
9107 00003CED 58                <1>       pop    eax
9108                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9109                            <1>       ;dec   dl
9110 00003CEE FEC0              <1>       inc    al  ; i++
9111 00003CF0 FECD              <1>       dec    ch
9112 00003CF2 75F1              <1>       jnz    short bfn_svs_1
9113                            <1>
9114 00003CF4 80E420            <1>       and    ah, 20h  ; (ar_index & 0x20)
9115                            <1>         ; for(i=0;i<=0x13;i++) {
9116 00003CF7 28C0              <1>       sub    al, al ; 0
9117 00003CF9 B514              <1>       mov    ch, 20
9118                            <1> bfn_svs_2:
9119                            <1>       ; inb(VGAREG_ACTL_RESET);
9120                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
9121 00003CFB B2DA              <1>       mov    dl, 0DAh
9122 00003CFD 50                <1>       push   eax
9123 00003CFE EC                <1>       in     al, dx
9124 00003CFF 8A0424            <1>       mov    al, [esp]
9125                            <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
9126 00003D02 08E0              <1>       or     al, ah
9127                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
9128 00003D04 B2C0              <1>       mov    dl, 0C0h
9129 00003D06 EE                <1>       out    dx, al
9130                            <1>       ;mov   dx, VGAREG_ACTL_READ_DATA ; 3C1h
9131                            <1>       ;mov   dl, 0C1h
9132 00003D07 FEC2              <1>       inc    dl
9133 00003D09 EC                <1>       in     al, dx
9134 00003D0A AA                <1>       stosb  ; (20 bytes in loop)
9135 00003D0B 58                <1>       pop    eax
9136 00003D0C FEC0              <1>       inc    al  ; i++
9137 00003D0E FECD              <1>       dec    ch
9138 00003D10 75E9              <1>       jnz    short bfn_svs_2
9139                            <1>
9140                            <1>       ; inb(VGAREG_ACTL_RESET);
9141                            <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
9142 00003D12 B2DA              <1>       mov    dl, 0DAh
9143 00003D14 EC                <1>       in     al, dx
9144                            <1>
9145                            <1>         ; for(i=0;i<=8;i++) {
9146 00003D15 28C0              <1>       sub    al, al ; 0
9147                            <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
9148                            <1>       ;mov   dl, 0CEh
9149 00003D17 B509              <1>       mov    ch, 9
9150                            <1> bfn_svs_3:
9151                            <1>         ; outb(VGAREG_GRDC_ADDRESS,i)
9152                            <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
9153 00003D19 B2CE              <1>       mov    dl, 0CEh
9154 00003D1B EE                <1>       out    dx, al
9155                            <1>       ; inb(VGAREG_ACTL_READ_DATA)
9156 00003D1C 50                <1>       push   eax
9157                            <1>       ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
9158                            <1>       ;mov   dl, 0CFh
9159 00003D1D FEC2              <1>       inc    dl
9160 00003D1F EC                <1>       in     al, dx
9161 00003D20 AA                <1>       stosb  ; (9 bytes in loop)
9162 00003D21 58                <1>       pop    eax
9163                            <1>       ;dec   dl
9164 00003D22 FEC0              <1>       inc    al  ; i++
9165 00003D24 FECD              <1>       dec    ch
9166 00003D26 75F1              <1>       jnz    short bfn_svs_3
9167                            <1>
9168                            <1>       ; write_word(ES, BX, crtc_addr); BX+= 2;
9169                            <1>       ; (offset 64)
9170 00003D28 66B8D403          <1>       mov    ax, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
9171 00003D2C 66AB              <1>       stosw  ; (2 bytes (1 word))
9172                            <1>
9173                            <1>         ; /* XXX: read plane latches */
9174 00003D2E 31C0              <1>       xor    eax, eax  ; 0
9175 00003D30 AB                <1>              stosd  ; (4 bytes)
9176                            <1>
9177                            <1>       ; (total 70 bytes are written above as controller hardware state)
9178                            <1>
9179                            <1> bfn_svs_4:
9180                            <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
9181 00003D31 F6C102            <1>       test   cl, 2
9182 00003D34 7476              <1>       jz     short bfn_svs_6
9183                            <1>
```

```
                                <1>       ; VIDEO BIOS DATA
                                <1>       ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
                                <1>       ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
                                <1>
                                <1>       ; if (CX & 2) {
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE)); BX++;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_NB_COLS)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS)); BX += 2;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS)); BX++;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT)); BX += 2;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL)); BX++;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES)); BX++;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL)); BX++;
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE)); BX += 2;
                                <1>         ;for(i=0;i<8;i++) {
                                <1>         ;   write_word(ES, BX, read_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i));
                                <1>         ;   BX += 2;
                                <1>         ;}
                                <1>         ;write_word(ES, BX, read_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START)); BX += 2;
                                <1>         ;write_byte(ES, BX, read_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE)); BX++;
                                <1>         ;/* current font */
                                <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(0, 0x1f * 4 + 2)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4)); BX += 2;
                                <1>         ;write_word(ES, BX, read_word(0, 0x43 * 4 + 2)); BX += 2;
                                <1>
                                <1>       ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
                                <1>       ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
                                <1>
00003D36 66B8D403               <1>       mov   ax, 3D4h ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
00003D3A 66AB                   <1>       stosw
00003D3C A0[DA6F0000]           <1>       mov   al, [CRT_MODE] ; Current video mode (0FFh for VESA VBE modes)
00003D41 AA                     <1>       stosb
00003D42 A0[DB6F0000]           <1>       mov   al, [CRT_MODE_SET] ; 29h for mode 03h ; TRDOS 386 feature only !
00003D47 AA                     <1>       stosb
00003D48 66A1[1E120300]         <1>       mov   ax, [video_mode] ; Current VESA VBE (SVGA, extended VGA) mode
00003D4E 66AB                   <1>       stosw                  ; (valid if [CRT_MODE] = 0FFh)
00003D50 66A1[B0960100]         <1>       mov   ax, [CRT_LEN] ; page size (in bytes)
00003D56 66AB                   <1>       stosw
00003D58 66A1[348A0100]         <1>       mov   ax, [CRT_START] ; video page start offset
00003D5E 66AB                   <1>       stosw
00003D60 A0[DC6F0000]           <1>       mov   al, [CRT_COLS] ; nbcols, characters per row
00003D65 AA                     <1>       stosb
00003D66 A0[E26F0000]           <1>       mov   al, [VGA_ROWS] ; nbrows, (character) rows per page (not rows-1)
00003D6B AA                     <1>       stosb
00003D6C A0[DE6F0000]           <1>       mov   al, [CHAR_HEIGHT] ; character font height (8 or 16 or 14)
00003D71 AA                     <1>       stosb
00003D72 A0[DF6F0000]           <1>       mov   al, [VGA_VIDEO_CTL] ; ROM BIOS DATA AREA Offset 87h
00003D77 AA                     <1>       stosb
00003D78 A0[E06F0000]           <1>       mov   al, [VGA_SWITCHES] ; feature bit switches
00003D7D AA                     <1>       stosb
00003D7E A0[E16F0000]           <1>       mov   al, [VGA_MODESET_CTL] ; basic mode set options
00003D83 AA                     <1>       stosb
                                <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
                                <1>       ; (bochs/plex86 does not use and return those)
00003D84 A0[DD6F0000]           <1>       mov   al, [CRT_PALETTE] ; current color palette ; TRDOS 386 feature only !
00003D89 AA                     <1>       stosb
00003D8A A0[468A0100]           <1>       mov   al, [ACTIVE_PAGE] ; current video page
00003D8F AA                     <1>       stosb
00003D90 66A1[F36F0000]         <1>       mov   ax, [CURSOR_MODE] ; cursor type
00003D96 66AB                   <1>       stosw
                                <1>       ;mov   eax, [CURSOR_POSN] ; cursor position for video page 0 and 1
                                <1>       ;stosd
                                <1>       ;mov   eax, [CURSOR_POSN+4] ; cursor position for video page 2 and 3
                                <1>       ;stosd
                                <1>       ;mov   eax, [CURSOR_POSN+8] ; cursor position for video page 4 and 5
                                <1>       ;stosd
                                <1>       ;mov   eax, [CURSOR_POSN+12] ; cursor position for video page 6 and 7
                                <1>       ;stosd
00003D98 56                     <1>       push  esi
00003D99 B504                   <1>       mov   ch, 4
00003D9B BE[368A0100]           <1>       mov   esi, CURSOR_POSN
                                <1> bfn_svs_5:
00003DA0 A5                     <1>       movsd
00003DA1 FECD                   <1>       dec   ch
00003DA3 75FB                   <1>       jnz   short bfn_svs_5
00003DA5 5E                     <1>       pop   esi
                                <1>       ; (font addr) protected mode address in kernel's/system memory space
                                <1>       ; (not accessable/meaningful address value by user)
00003DA6 A1[C2960100]           <1>       mov   eax, [VGA_INT43H] ; VGA current (default) font address
00003DAB AB                     <1>       stosd
                                <1>
                                <1>       ; (total 40 bytes are written above as BIOS data state)
                                <1>
                                <1> bfn_svs_6:
                                <1>       ; 12/01/2021
00003DAC F6C104                 <1>       test  cl, 4
00003DAF 7423                   <1>       jz    short bfn_svs_8
                                <1>
                                <1>       ;/* XXX: check this */
                                <1>             ; /* read/write mode dac */
                                <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_STATE)); BX++;
                                <1>       ;     ; /* pix address */
                                <1>         ;write_byte(ES, BX, inb(VGAREG_DAC_WRITE_ADDRESS)); BX++;
                                <1>         ;write_byte(ES, BX, inb(VGAREG_PEL_MASK)); BX++;
                                <1>         ;// Set the whole dac always, from 0
                                <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
                                <1>         ;for(i=0;i<256*3;i++) {
                                <1>         ;   write_byte(ES, BX, inb(VGAREG_DAC_DATA)); BX++;
                                <1>         ;}
                                <1>         ;write_byte(ES, BX, 0); BX++; /* color select register */
                                <1>
                                <1>       ; /* read/write mode dac */
00003DB1 66BAC703               <1>       mov   dx, 3C7h ; VGAREG_DAC_STATE
```

```
9289 00003DB5 EC                <1>       in     al, dx
9290 00003DB6 AA                <1>       stosb
9291                            <1>       ; /* pix address */
9292                            <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9293                            <1>       ;mov   dl, 0C8h
9294 00003DB7 FEC2              <1>       inc    dl
9295 00003DB9 EC                <1>       in     al, dx
9296 00003DBA AA                <1>       stosb
9297                            <1>       ;mov   dx, VGAREG_PEL_MASK  ; 3C6h
9298 00003DBB B2C6              <1>       mov    dl, 0C6h
9299 00003DBD EC                <1>       in     al, dx
9300 00003DBE AA                <1>       stosb
9301                            <1>       ;// Set the whole dac always, from 0
9302 00003DBF 30C0              <1>       xor    al, al ; 0
9303                            <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9304 00003DC1 B2C8              <1>       mov    dl, 0C8h
9305 00003DC3 EE                <1>       out    dx, al
9306                            <1>
9307 00003DC4 51                <1>       push   ecx ; 22/01/2021
9308                            <1>       ;for(i=0;i<256*3;i++) {
9309 00003DC5 B900030000        <1>       mov    ecx, 256*3 ; 768 bytes
9310                            <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
9311                            <1>       ;mov   dl, 0C9h
9312 00003DCA FEC2              <1>       inc    dl ; dx = 3C9h
9313                            <1> bfn_svs_7:
9314 00003DCC EC                <1>       in     al, dx
9315 00003DCD AA                <1>       stosb
9316 00003DCE E2FC              <1>       loop   bfn_svs_7
9317 00003DD0 59                <1>       pop    ecx ; 22/01/2021
9318                            <1>
9319                            <1>       ; /* color select register */
9320 00003DD1 28C0              <1>       sub    al, al ; 0
9321 00003DD3 AA                <1>       stosb
9322                            <1>
9323                            <1>       ; (total 772 bytes are written above as DAC state)
9324                            <1> bfn_svs_8:
9325 00003DD4 C3                <1>       retn
9326                            <1>
9327                            <1> vbe_biosfn_save_video_state:
9328                            <1>       ; 23/01/2021
9329                            <1>       ; 13/01/2021
9330                            <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
9331                            <1>       ; (vbe.c)
9332                            <1>
9333                            <1>       ; modified registers: eax, edx, edi, ch
9334                            <1>
9335                            <1>       ; input: edi = state buffer address
9336                            <1>       ; output:
9337                            <1>       ;      VBE DISPI register contents will be saved
9338                            <1>       ;      (18 bytes, 9 words)
9339                            <1>
9340                            <1>       ; outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
9341                            <1>       ; enable = inw(VBE_DISPI_IOPORT_DATA);
9342                            <1>       ; write_word(ES, BX, enable);
9343                            <1>       ; BX += 2;
9344                            <1>       ; if (!(enable & VBE_DISPI_ENABLED))
9345                            <1>       ;     return;
9346                            <1>       ; for(i = VBE_DISPI_INDEX_XRES;
9347                            <1>       ;             i <= VBE_DISPI_INDEX_Y_OFFSET; i++) {
9348                            <1>       ;    if (i != VBE_DISPI_INDEX_ENABLE) {
9349                            <1>       ;        outw(VBE_DISPI_IOPORT_INDEX, i);
9350                            <1>       ;        write_word(ES, BX, inw(VBE_DISPI_IOPORT_DATA));
9351                            <1>         ;    BX += 2;
9352                            <1>       ;        }
9353                            <1>       ; }
9354                            <1>
9355 00003DD5 66BACE01          <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
9356 00003DD9 B804000000        <1>       mov    eax, 04h  ; VBE_DISPI_INDEX_ENABLE
9357 00003DDE 66EF              <1>       out    dx, ax
9358                            <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
9359 00003DE0 FEC2              <1>       inc    dl
9360 00003DE2 66ED              <1>       in     ax, dx ; enable (status)
9361 00003DE4 66AB              <1>       stosw
9362 00003DE6 6683E001          <1>       and    ax, 1 ; VBE_DISPI_ENABLED
9363 00003DEA 7505              <1>       jnz    short vbe_bfn_svs_0
9364                            <1>       ; 23/01/2021
9365                            <1>       ; ax = 0
9366                            <1>       ; VBE_DISPI_DISABLED
9367                            <1>       ; 13/01/2021
9368                            <1>       ; clear remain 8 bytes
9369                            <1>       ;xor   eax, eax
9370 00003DEC AB                <1>       stosd  ; 2
9371 00003DED AB                <1>       stosd  ; 2
9372 00003DEE AB                <1>       stosd  ; 2
9373 00003DEF AB                <1>       stosd  ; 2
9374 00003DF0 C3                <1>       retn
9375                            <1> vbe_bfn_svs_0:
9376                            <1>       ; VBE_DISPI_ENABLED
9377                            <1>
9378                            <1>       ;sub   eax, eax
9379 00003DF1 28C0              <1>       sub    al, al ; eax = 0
9380                            <1>
9381                            <1>       ; from VBE_DISPI_INDEX_XRES
9382                            <1>       ;   to VBE_DISPI_INDEX_BPP
9383                            <1>
9384 00003DF3 B503              <1>       mov    ch, 3
9385                            <1>       ; al = 0 ; VBE_DISPI_INDEX_XRES - 1
9386                            <1>
9387 00003DF5 E804000000        <1>       call   vbe_bfn_svs_1
9388                            <1>
9389                            <1>       ; from VBE_DISPI_INDEX_BANK
9390                            <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
9391                            <1>
9392 00003DFA FEC0              <1>       inc    al
9393                            <1>       ; al = 4 ; VBE_DISPI_INDEX_BANK - 1
```

```
                                <1>
00003DFC B505                   <1>       mov    ch, 5
                                <1> vbe_bfn_svs_1:
00003DFE FEC0                   <1>       inc    al ; from VBE_DISPI_INDEX_XRES
                                <1>                 ;   to VBE_DISPI_INDEX_BPP
                                <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003E00 FECA                   <1>       dec    dl ; 1CEh
00003E02 66EF                   <1>       out    dx, ax
00003E04 50                     <1>       push   eax
                                <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
00003E05 FEC2                   <1>       inc    dl ; 1CFh
00003E07 66ED                   <1>       in     ax, dx
00003E09 66AB                   <1>       stosw
00003E0B 58                     <1>       pop    eax
00003E0C FECD                   <1>       dec    ch
00003E0E 75EE                   <1>       jnz    short vbe_bfn_svs_1
00003E10 C3                     <1>       retn
                                <1>
                                <1> ; 11/01/2021
                                <1> VGAREG_ACTL_ADDRESS       equ 3C0h
                                <1> VGAREG_ACTL_WRITE_DATA    equ 3C0h
                                <1> VGAREG_ACTL_READ_DATA     equ 3C1h
                                <1>
                                <1> VGAREG_INPUT_STATUS       equ 3C2h
                                <1> VGAREG_WRITE_MISC_OUTPUT equ 3C2h
                                <1> VGAREG_VIDEO_ENABLE       equ 3C3h
                                <1> VGAREG_SEQU_ADDRESS       equ 3C4h
                                <1> VGAREG_SEQU_DATA   equ 3C5h
                                <1>
                                <1> VGAREG_PEL_MASK           equ 3C6h
                                <1> VGAREG_DAC_STATE   equ 3C7h
                                <1> VGAREG_DAC_READ_ADDRESS   equ 3C7h
                                <1> VGAREG_DAC_WRITE_ADDRESS equ 3C8h
                                <1> VGAREG_DAC_DATA           equ 3C9h
                                <1>
                                <1> VGAREG_READ_FEATURE_CTL   equ 3CAh
                                <1> VGAREG_READ_MISC_OUTPUT   equ 3CCh
                                <1>
                                <1> VGAREG_GRDC_ADDRESS       equ 3CEh
                                <1> VGAREG_GRDC_DATA   equ 3CFh
                                <1>
                                <1> ;VGAREG_MDA_CRTC_ADDRESS equ 3B4h
                                <1> ;VGAREG_MDA_CRTC_DATA     equ 3B5h
                                <1> VGAREG_VGA_CRTC_ADDRESS   equ 3D4h
                                <1> VGAREG_VGA_CRTC_DATA      equ 3D5h
                                <1>
                                <1> ;VGAREG_MDA_WRITE_FEATURE_CTL equ 3BAh
                                <1> VGAREG_VGA_WRITE_FEATURE_CTL equ 3DAh
                                <1> VGAREG_ACTL_RESET  equ 3DAh
                                <1>
                                <1> ;VGAREG_MDA_MODECTL       equ 3B8h
                                <1> VGAREG_CGA_MODECTL equ 3D8h
                                <1> VGAREG_CGA_PALETTE equ 3D9h
                                <1>
                                <1> biosfn_restore_video_state:
                                <1>       ; 22/01/2021
                                <1>       ; 13/01/2021
                                <1>       ; 12/01/2021 (TRDOS 386 v2.0.3)
                                <1>       ; (vgabios.c)
                                <1>
                                <1>       ; modified registers: eax, edx, esi, edi, ch
                                <1>
                                <1>       ;mov   esi, VBE3SAVERESTOREBLOCK
                                <1>
                                <1>       ; input: esi = state buffer address
                                <1>
00003E11 F6C101                 <1>       test   cl, 1
00003E14 0F84A9000000           <1>       jz     bfn_rvs_6
                                <1>
00003E1A 66817E40D403           <1>       cmp    word [esi+64], 3D4h ; must be 3D4h
00003E20 7402                   <1>       je     short bfn_rvs_0
                                <1>                    ; it is seen as valid buffer
00003E22 F9                     <1>       stc
00003E23 C3                     <1>       retn
                                <1>
                                <1> bfn_rvs_0:
00003E24 89F7                   <1>       mov    edi, esi ; addr1
00003E26 83C605                 <1>       add    esi, 5 ; skip 1st 5 bytes for now
                                <1>
                                <1>       ; // Reset Attribute Ctl flip-flop
                                <1>         ; inb(VGAREG_ACTL_RESET);
00003E29 66BADA03               <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
00003E2D EC                     <1>       in     al, dx
                                <1>
                                <1>       ; for(i=1;i<=4;i++){
00003E2E B001                   <1>       mov    al, 1
                                <1>       ;;mov  dx, VGAREG_SEQU_ADDRESS ; 3C4h
                                <1>       ;mov   dl, 0C4h
00003E30 B504                   <1>       mov    ch, 4
                                <1> bfn_rvs_1:
                                <1>       ; outb(VGAREG_SEQU_ADDRESS, i);
                                <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
00003E32 B2C4                   <1>       mov    dl, 0C4h
00003E34 EE                     <1>       out    dx, al
                                <1>       ;mov   dx, VGAREG_SEQU_DATA  ; 3C5h
00003E35 FEC2                   <1>       inc    dl  ; dx = 3C5h
                                <1>       ; outb(VGAREG_SEQU_DATA)
00003E37 50                     <1>       push   eax
00003E38 AC                     <1>       lodsb  ; (4 bytes in loop)
00003E39 EE                     <1>       out    dx, al
00003E3A 58                     <1>       pop    eax
                                <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C4h
                                <1>       ;dec   dl
00003E3B FEC0                   <1>       inc    al  ; i++
00003E3D FECD                   <1>       dec    ch
```

```
9499 00003E3F 75F1              <1>       jnz    short bfn_rvs_1
9500                            <1>
9501                            <1>       ; outb(VGAREG_SEQU_ADDRESS, 0);
9502 00003E41 28C0              <1>       sub    al, al ; 0
9503 00003E43 EE                <1>       out    dx, al
9504                            <1>       ; outb(VGAREG_SEQU_DATA)
9505                            <1>       ;mov   dx, VGAREG_SEQU_DATA ; 3C5h
9506 00003E44 FEC2              <1>       inc    dl  ; dx = 3C5h
9507 00003E46 AC                <1>       lodsb  ; (+1 byte)
9508 00003E47 EE                <1>       out    dx, al
9509                            <1>
9510                            <1>       ; // Disable CRTC write protection
9511                            <1>       ; outw(crtc_addr,0x0011);
9512                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9513 00003E48 B2D4              <1>       mov    dl, 0D4h
9514 00003E4A 66B81100          <1>       mov    ax, 11h
9515 00003E4E 66EF              <1>       out    dx, ax
9516                            <1>
9517                            <1>       ; // Set CRTC regs
9518                            <1>
9519                            <1>       ; for(i=0;i<=0x18;i++) {
9520                            <1>         ;   if (i != 0x11) {
9521 00003E50 28C0              <1>       sub    al, al ; 0
9522                            <1>       ;;mov  dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9523                            <1>       ;mov   dl, 0D4h
9524 00003E52 B519              <1>       mov    ch, 25
9525                            <1> bfn_rvs_2:
9526                            <1>         ; outb(crtc_addr,i);
9527                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9528 00003E54 B2D4              <1>       mov    dl, 0D4h
9529 00003E56 EE                <1>       out    dx, al
9530                            <1>       ;mov   dx, VGAREG_VGA_CRTC_DATA ; 3D5h
9531 00003E57 FEC2              <1>       inc    dl  ; dx = 3D5h
9532                            <1>       ; inb(crtc_addr+1)
9533 00003E59 50                <1>       push   eax
9534 00003E5A AC                <1>       lodsb  ; (25 bytes in loop)
9535 00003E5B EE                <1>       out    dx, al
9536 00003E5C 58                <1>       pop    eax
9537                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9538                            <1>       ;dec   dl
9539 00003E5D FEC0              <1>       inc    al  ; i++
9540 00003E5F 3C11              <1>       cmp    al, 17 ; 11h
9541 00003E61 7505              <1>       jne    short bfn_rvs_3
9542 00003E63 AC                <1>       lodsb
9543 00003E64 88C4              <1>       mov    ah, al ; *
9544 00003E66 B012              <1>       mov    al, 18
9545                            <1> bfn_rvs_3:
9546 00003E68 FECD              <1>       dec    ch
9547 00003E6A 75E8              <1>       jnz    short bfn_rvs_2
9548                            <1>
9549                            <1>       ; // select crtc base address
9550                            <1>         ; v = inb(VGAREG_READ_MISC_OUTPUT) & ~0x01;
9551                            <1>       ;if (crtc_addr = 0x3d4)
9552                            <1>       ;   v |= 0x01;
9553                            <1>       ; outb(VGAREG_WRITE_MISC_OUTPUT, v);
9554                            <1>
9555                            <1>       ;;mov  dx, VGAREG_READ_MISC_OUTPUT ; 3CCh
9556                            <1>       ;mov   dl, 0CCh
9557                            <1>       ;in    al, dl
9558                            <1>       ;and   al, 1
9559                            <1>       ;;mov  dx, VGAREG_WRITE_MISC_OUTPUT ; 3C2h
9560                            <1>       ;mov   dl, 0C2h
9561                            <1>       ;or    al, 1
9562                            <1>       ;out   dx, al
9563                            <1>
9564                            <1>       ; // enable write protection if needed
9565                            <1>       ;outb(crtc_addr, 0x11);
9566                            <1>         ;outb(crtc_addr+1, read_byte(ES, BX - 0x18 + 0x11));
9567                            <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9568 00003E6C B2D4              <1>       mov    dl, 0D4h
9569 00003E6E B011              <1>       mov    al, 11h
9570 00003E70 EE                <1>       out    dx, al
9571 00003E71 88E0              <1>       mov    al, ah ; *
9572 00003E73 FEC2              <1>       inc    dl ; dx = 3D5h
9573 00003E75 EE                <1>       out    dx, al
9574                            <1>
9575                            <1>       ; // Set Attribute Ctl
9576 00003E76 8A6703            <1>       mov    ah, [edi+3] ; addr1+3, ah = ar_index
9577 00003E79 80E420            <1>       and    ah, 20h  ; (ar_index & 0x20)
9578                            <1>
9579                            <1>         ; inb(VGAREG_ACTL_RESET);
9580                            <1>       ;mov   dx, 3DAh ; VGAREG_ACTL_RESET
9581 00003E7C B2DA              <1>       mov    dl, 0DAh
9582 00003E7E EC                <1>       in     al, dx
9583                            <1>
9584                            <1>         ; for(i=0;i<=0x13;i++) {
9585 00003E7F 28C0              <1>       sub    al, al ; 0
9586 00003E81 B514              <1>       mov    ch, 20
9587                            <1> bfn_rvs_4:
9588                            <1>       ; outb(VGAREG_ACTL_ADDRESS, i | (ar_index & 0x20));
9589 00003E83 50                <1>       push   eax
9590 00003E84 08E0              <1>       or     al, ah
9591                            <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
9592 00003E86 B2C0              <1>       mov    dl, 0C0h
9593 00003E88 EE                <1>       out    dx, al
9594                            <1>       ;mov   dx, VGAREG_ACTL_WRITE_DATA ; 3C0h
9595                            <1>       ;mov   dl, 0C0h
9596 00003E89 AC                <1>       lodsb  ; (20 bytes in loop)
9597 00003E8A EE                <1>       out    dx, al
9598 00003E8B 58                <1>       pop    eax
9599 00003E8C FEC0              <1>       inc    al  ; i++
9600 00003E8E FECD              <1>       dec    ch
9601 00003E90 75F1              <1>       jnz    short bfn_rvs_4
9602                            <1>
9603                            <1>       ; outb(VGAREG_ACTL_ADDRESS, ar_index);
```

```
9604                                    <1>       ;mov   dx, VGAREG_ACTL_ADDRESS ; 3C0h
9605                                    <1>       ;mov   dl, 0C0h
9606 00003E92 88E0                      <1>       mov    al, ah ; ar_index
9607 00003E94 EE                        <1>       out    dx, al
9608                                    <1>
9609                                    <1>       ; inb(VGAREG_ACTL_RESET);
9610                                    <1>       ;mov   dx, VGAREG_ACTL_RESET ; 3DAh
9611 00003E95 B2DA                      <1>       mov    dl, 0DAh
9612 00003E97 EC                        <1>       in     al, dx
9613                                    <1>
9614                                    <1>         ; for(i=0;i<=8;i++) {
9615 00003E98 28C0                      <1>       sub    al, al ; 0
9616                                    <1>       ;;mov  dx, VGAREG_GRDC_ADDRESS ; 3CEh
9617                                    <1>       ;mov   dl, 0CEh
9618 00003E9A B509                      <1>       mov    ch, 9
9619                                    <1> bfn_rvs_5:
9620                                    <1>       ; outb(VGAREG_GRDC_ADDRESS,i)
9621                                    <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
9622 00003E9C B2CE                      <1>       mov    dl, 0CEh
9623 00003E9E EE                        <1>       out    dx, al
9624                                    <1>       ; outb(VGAREG_ACTL_READ_DATA)
9625 00003E9F 50                        <1>       push   eax
9626                                    <1>       ;mov   dx, VGAREG_GRDC_DATA ; 3CFh
9627                                    <1>       ;mov   dl, 0CFh
9628 00003EA0 FEC2                      <1>       inc    dl
9629 00003EA2 AC                        <1>       lodsb  ; (9 bytes in loop)
9630 00003EA3 EE                        <1>       out    dx, al
9631 00003EA4 58                        <1>       pop    eax
9632                                    <1>       ;dec   dl
9633 00003EA5 FEC0                      <1>       inc    al  ; i++
9634 00003EA7 FECD                      <1>       dec    ch
9635 00003EA9 75F1                      <1>       jnz    short bfn_rvs_5
9636                                    <1>
9637                                    <1>       ; BX += 2; /* crtc_addr */     ; 3D4h
9638                                    <1>         ; BX += 4; /* plane latches */ ; 0
9639 00003EAB 83C606                    <1>       add    esi, 6
9640 00003EAE 56                        <1>       push   esi ; *
9641                                    <1>
9642                                    <1>       ;outb(VGAREG_SEQU_ADDRESS, read_byte(ES, addr1)); addr1++;
9643                                    <1>         ;outb(crtc_addr, read_byte(ES, addr1)); addr1++;
9644                                    <1>         ;outb(VGAREG_GRDC_ADDRESS, read_byte(ES, addr1)); addr1++;
9645                                    <1>         ;addr1++;
9646                                    <1>         ;outb(crtc_addr - 0x4 + 0xa, read_byte(ES, addr1)); addr1++;
9647                                    <1>
9648 00003EAF 89FE                      <1>       mov    esi, edi ; start of state buffer
9649                                    <1>
9650                                    <1>       ;mov   dx, VGAREG_SEQU_ADDRESS ; 3C7h
9651 00003EB1 B2C7                      <1>       mov    dl, 0C7h
9652 00003EB3 AC                        <1>       lodsb
9653 00003EB4 EE                        <1>       out    dx, al
9654                                    <1>       ;mov   dx, VGAREG_VGA_CRTC_ADDRESS ; 3D4h
9655 00003EB5 B2D4                      <1>       mov    dl, 0D4h
9656 00003EB7 AC                        <1>       lodsb
9657 00003EB8 EE                        <1>       out    dx, al
9658                                    <1>       ;mov   dx, VGAREG_GRDC_ADDRESS ; 3CEh
9659 00003EB9 B2CE                      <1>         mov  dl, 0CEh
9660 00003EBB AC                        <1>       lodsb
9661 00003EBC EE                        <1>       out    dx, al
9662 00003EBD AC                        <1>       lodsb  ; addr1++
9663                                    <1>       ;mov   dx, VGAREG_VGA_WRITE_FEATURE_CTL ; 3DAh
9664 00003EBE B2DA                      <1>       mov    dl, 0DAh
9665 00003EC0 AC                        <1>       lodsb
9666 00003EC1 EE                        <1>       out    dx, al
9667                                    <1>
9668 00003EC2 5E                        <1>       pop    esi ; *
9669                                    <1>
9670                                    <1>       ; (total 70 bytes are read above as controller hardware state)
9671                                    <1>
9672                                    <1> bfn_rvs_6:
9673                                    <1>       ; 13/01/2021
9674 00003EC3 F6C102                    <1>       test   cl, 2
9675 00003EC6 747D                      <1>       jz     short bfn_rvs_9
9676                                    <1>
9677                                    <1>       ; VIDEO BIOS DATA
9678                                    <1>       ; !!! this data is valid for TRDOS 386 v2 kernel only !!!
9679                                    <1>       ; (this is not same with BOCHS/PLEX86 video bios, BIOS data)
9680                                    <1>
9681                                    <1>       ; if (CX & 2) {
9682                                    <1>       ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_MODE, read_byte(ES, BX)); BX++;
9683                                    <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_NB_COLS, read_word(ES, BX)); BX += 2;
9684                                    <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_PAGE_SIZE, read_word(ES, BX)); BX += 2;
9685                                    <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CRTC_ADDRESS, read_word(ES, BX)); BX += 2;
9686                                    <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_NB_ROWS, read_byte(ES, BX)); BX++;
9687                                    <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CHAR_HEIGHT, read_word(ES, BX)); BX += 2;
9688                                    <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_VIDEO_CTL, read_byte(ES, BX)); BX++;
9689                                    <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_SWITCHES, read_byte(ES, BX)); BX++;
9690                                    <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_MODESET_CTL, read_byte(ES, BX)); BX++;
9691                                    <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CURSOR_TYPE, read_word(ES, BX)); BX += 2;
9692                                    <1>         ;for(i=0;i<8;i++) {
9693                                    <1>         ;   write_word(BIOSMEM_SEG, BIOSMEM_CURSOR_POS+2*i, read_word(ES, BX));
9694                                    <1>         ;   BX += 2;
9695                                    <1>         ;}
9696                                    <1>         ;write_word(BIOSMEM_SEG,BIOSMEM_CURRENT_START, read_word(ES, BX)); BX += 2;
9697                                    <1>         ;write_byte(BIOSMEM_SEG,BIOSMEM_CURRENT_PAGE, read_byte(ES, BX)); BX++;
9698                                    <1>         ;/* current font */
9699                                    <1>         ;write_word(0, 0x1f * 4, read_word(ES, BX)); BX += 2;
9700                                    <1>         ;write_word(0, 0x1f * 4 + 2, read_word(ES, BX)); BX += 2;
9701                                    <1>         ;write_word(0, 0x43 * 4, read_word(ES, BX)); BX += 2;
9702                                    <1>         ;write_word(0, 0x43 * 4 + 2, read_word(ES, BX)); BX += 2;
9703                                    <1>
9704                                    <1>       ; !!! save TRDOS 386 v2 kernel spesific video bios data !!!
9705                                    <1>       ; (which is/are used by 'SET_MODE' function and/or it's sub functions)
9706                                    <1>
9707 00003EC8 66AD                      <1>       lodsw   ; CRTC_ADDR, always 3D4h (color VGA) for TRDOS 386 v2
9708                                    <1>       ; skip 3D4h check if it is already checked
```

```
9709 00003ECA F6C101              <1>       test   cl, 1
9710 00003ECD 7508                <1>       jnz    short bfn_rvs_7
9711 00003ECF 663DD403            <1>       cmp    ax, 3D4h
9712 00003ED3 7402                <1>       je     short bfn_rvs_7
9713 00003ED5 F9                  <1>       stc
9714 00003ED6 C3                  <1>       retn
9715                              <1> bfn_rvs_7:
9716 00003ED7 AC                  <1>       lodsb
9717 00003ED8 A2[DA6F0000]        <1>       mov    [CRT_MODE], al ; Current video mode (0FFh for VESA VBE modes)
9718 00003EDD AC                  <1>       lodsb
9719 00003EDE A2[DB6F0000]        <1>       mov    [CRT_MODE_SET], al ; 29h for mode 03h ; TRDOS 386 feature only !
9720 00003EE3 66AD                <1>       lodsw
9721 00003EE5 66A3[1E120300]      <1>       mov    [video_mode], ax ; Current VESA VBE (SVGA, extended VGA) mode
9722 00003EEB 66AD                <1>       lodsw          ; (valid if [CRT_MODE] = 0FFh)
9723 00003EED 66A3[B0960100]      <1>       mov    [CRT_LEN], ax ; page size (in bytes)
9724 00003EF3 66AD                <1>       lodsw
9725 00003EF5 66A3[348A0100]      <1>       mov    [CRT_START], ax ; video page start offset
9726 00003EFB AC                  <1>       lodsb
9727 00003EFC A2[DC6F0000]        <1>       mov    [CRT_COLS], al ; nbcols, characters per row
9728 00003F01 AC                  <1>       lodsb
9729 00003F02 A2[E26F0000]        <1>       mov    [VGA_ROWS], al ; nbrows, (character) rows per page (not rows-1)
9730 00003F07 AC                  <1>       lodsb
9731 00003F08 A2[DE6F0000]        <1>       mov    [CHAR_HEIGHT], al ; character font height (8 or 16 or 14)
9732 00003F0D AC                  <1>       lodsb
9733 00003F0E A2[DF6F0000]        <1>       mov    [VGA_VIDEO_CTL], al ; ROM BIOS DATA AREA Offset 87h
9734 00003F13 AC                  <1>       lodsb
9735 00003F14 A2[E06F0000]        <1>       mov    [VGA_SWITCHES], al ; feature bit switches
9736 00003F19 AC                  <1>       lodsb
9737 00003F1A A2[E16F0000]        <1>       mov    [VGA_MODESET_CTL], al ; basic mode set options
9738                              <1>       ; followings are only used by TRDOS 386 v2 (IBM PC/AT ROMBIOS) code
9739                              <1>       ; (bochs/plex86 does not use and return those)
9740 00003F1F AC                  <1>       lodsb
9741 00003F20 A2[DD6F0000]        <1>       mov    [CRT_PALETTE], al ; current color palette ; TRDOS 386 feature only !
9742 00003F25 AC                  <1>       lodsb
9743 00003F26 A2[468A0100]        <1>       mov    [ACTIVE_PAGE], al ; current video page
9744 00003F2B 66AD                <1>       lodsw
9745 00003F2D 66A3[F36F0000]      <1>       mov    [CURSOR_MODE], ax ; cursor type
9746                              <1>       ;lodsd
9747                              <1>       ;mov   [CURSOR_POSN], eax ; cursor position for video page 0 and 1
9748                              <1>       ;lodsd
9749                              <1>       ;mov   [CURSOR_POSN+4], eax ; cursor position for video page 2 and 3
9750                              <1>       ;lodsd
9751                              <1>       ;mov   [CURSOR_POSN+8], eax ; cursor position for video page 4 and 5
9752                              <1>       ;lodsd
9753                              <1>       ;mov   [CURSOR_POSN+12], eax ; cursor position for video page 6 and 7
9754 00003F33 B504                <1>       mov    ch, 4
9755 00003F35 BF[368A0100]        <1>       mov    edi, CURSOR_POSN
9756                              <1> bfn_rvs_8:
9757 00003F3A A5                  <1>       movsd
9758 00003F3B FECD                <1>       dec    ch
9759 00003F3D 75FB                <1>       jnz    short bfn_rvs_8
9760                              <1>       ; (font addr) protected mode address in kernel's/system memory space
9761                              <1>       ; (not accessable/meaningful address value by user)
9762 00003F3F AD                  <1>       lodsd
9763 00003F40 A3[C2960100]        <1>       mov    [VGA_INT43H], eax ; VGA current (default) font address
9764                              <1>
9765                              <1>       ; (total 40 bytes are read&written above as BIOS data state)
9766                              <1> bfn_rvs_9:
9767                              <1>       ; 13/01/2021
9768 00003F45 F6C104              <1>       test   cl, 4
9769 00003F48 7422                <1>       jz     short bfn_rvs_11
9770                              <1>
9771                              <1>       ;BX++;
9772                              <1>         ;v = read_byte(ES, BX); BX++;
9773                              <1>         ;outb(VGAREG_PEL_MASK, read_byte(ES, BX)); BX++;
9774                              <1>         ;// Set the whole dac always, from 0
9775                              <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS,0x00);
9776                              <1>         ;for(i=0;i<256*3;i++) {
9777                              <1>         ;   outb(VGAREG_DAC_DATA, read_byte(ES, BX)); BX++;
9778                              <1>         ;}
9779                              <1>         ;BX++;
9780                              <1>         ;outb(VGAREG_DAC_WRITE_ADDRESS, v);
9781                              <1>
9782                              <1>       ; /* read/write mode dac */
9783 00003F4A AC                  <1>       lodsb  ; skip ; VGAREG_DAC_STATE
9784 00003F4B AC                  <1>       lodsb
9785 00003F4C 88C4                <1>       mov    ah, al ; * ; v
9786 00003F4E AC                  <1>       lodsb
9787 00003F4F 66BAC603            <1>       mov    dx, VGAREG_PEL_MASK  ; 3C6h
9788 00003F53 EE                  <1>       out    dx, al
9789                              <1>       ;// Set the whole dac always, from 0
9790 00003F54 30C0                <1>       xor    al, al ; 0
9791                              <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9792 00003F56 B2C8                <1>       mov    dl, 0C8h
9793 00003F58 EE                  <1>       out    dx, al
9794                              <1>
9795 00003F59 51                  <1>       push   ecx ; 22/01/2021
9796                              <1>       ;for(i=0;i<256*3;i++) {
9797 00003F5A B900030000          <1>       mov    ecx, 256*3 ; 768 bytes
9798                              <1>       ;mov   dx, VGAREG_DAC_DATA ; 3C9h
9799                              <1>       ;mov   dl, 0C9h
9800 00003F5F FEC2                <1>       inc    dl ; dx = 3C9h
9801                              <1> bfn_rvs_10:
9802 00003F61 AC                  <1>       lodsb
9803 00003F62 EE                  <1>       out    dx, al
9804 00003F63 E2FC                <1>       loop   bfn_rvs_10
9805 00003F65 59                  <1>       pop    ecx ; 22/01/2021
9806                              <1>
9807                              <1>       ; /* color select register */
9808 00003F66 AC                  <1>       lodsb   ; skip
9809                              <1>
9810 00003F67 88E0                <1>       mov    al, ah ; * ; v
9811                              <1>
9812                              <1>       ;mov   dx, VGAREG_DAC_WRITE_ADDRESS ; 3C8h
9813                              <1>       ;mov   dl, 0C8h
```

```
9814 00003F69 FECA              <1>       dec    dl ; dx  = 3C8h
9815 00003F6B EE                <1>       out    dx, al ; * ; v
9816                            <1>
9817                            <1>       ; (total 772 bytes are read above as DAC state)
9818                            <1> bfn_rvs_11:
9819 00003F6C C3                <1>       retn
9820                            <1>
9821                            <1> vbe_biosfn_restore_video_state:
9822                            <1>       ; 23/01/2021
9823                            <1>       ; 13/01/2021 (TRDOS 386 v2.0.3)
9824                            <1>       ; (vbe.c)
9825                            <1>
9826                            <1>       ; modified registers: eax, edx, esi, ch
9827                            <1>
9828                            <1>       ; input: esi = state buffer address
9829                            <1>       ; output:
9830                            <1>       ;       VBE DISPI register contents will be restored
9831                            <1>       ;       (18 bytes, 9 words)
9832                            <1>
9833                            <1>       ; enable = read_word(ES, BX);
9834                            <1>       ; BX += 2;
9835                            <1>       ;
9836                            <1>       ; if (!(enable & VBE_DISPI_ENABLED)) {
9837                            <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
9838                            <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
9839                            <1>       ; } else {
9840                            <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_XRES);
9841                            <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
9842                            <1>         ;   BX += 2;
9843                            <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_YRES);
9844                            <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
9845                            <1>         ;   BX += 2;
9846                            <1>         ;   outw(VBE_DISPI_IOPORT_INDEX, VBE_DISPI_INDEX_BPP);
9847                            <1>         ;   outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
9848                            <1>         ;   BX += 2;
9849                            <1>         ;   outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
9850                            <1>         ;   outw(VBE_DISPI_IOPORT_DATA, enable);
9851                            <1>       ;
9852                            <1>         ;  for(i = VBE_DISPI_INDEX_BANK; i <= VBE_DISPI_INDEX_Y_OFFSET; i++)
9853                            <1>       ;    {
9854                            <1>         ;     outw(VBE_DISPI_IOPORT_INDEX, i);
9855                            <1>         ;     outw(VBE_DISPI_IOPORT_DATA, read_word(ES, BX));
9856                            <1>         ;     BX += 2;
9857                            <1>         ;    }
9858                            <1>         ; }
9859                            <1>
9860 00003F6D 66AD              <1>       lodsw  ; enable (status, enabled=1, disabled=0)
9861 00003F6F 66BACE01          <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
9862                            <1>       ; 23/01/2021
9863 00003F73 6683E001          <1>       and    ax, 1 ; VBE_DISPI_ENABLED
9864 00003F77 750B              <1>       jnz    short vbe_bfn_rvs_1
9865                            <1>       ; ax = 0
9866                            <1>       ; VBE_DISPI_DISABLED
9867                            <1> vbe_bfn_rvs_0:
9868                            <1>       ; enable (disable) dispi
9869                            <1>       ; dx = 01CEh ; VBE_DISPI_IOPORT_INDEX
9870                            <1>       ; ah = 0
9871 00003F79 50                <1>       push   eax
9872 00003F7A B004              <1>       mov    al, 04h      ; VBE_DISPI_INDEX_ENABLE
9873 00003F7C 66EF              <1>       out    dx, ax
9874                            <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
9875 00003F7E FEC2              <1>       inc    dl
9876 00003F80 58                <1>       pop    eax
9877 00003F81 66EF              <1>       out    dx, ax ; enable (or disable)
9878 00003F83 C3                <1>       retn
9879                            <1> vbe_bfn_rvs_1:
9880                            <1>       ; VBE_DISPI_ENABLED
9881                            <1>
9882                            <1>       ; from VBE_DISPI_INDEX_XRES
9883                            <1>       ;   to VBE_DISPI_INDEX_BPP
9884                            <1>
9885 00003F84 B503              <1>       mov    ch, 3
9886 00003F86 28C0              <1>       sub    al, al ; 0 ; VBE_DISPI_INDEX_XRES - 1
9887                            <1>       ; ax = 0
9888                            <1>
9889 00003F88 E80B000000        <1>       call   vbe_bfn_rvs_2
9890                            <1>
9891                            <1>         ;outw(VBE_DISPI_IOPORT_INDEX,VBE_DISPI_INDEX_ENABLE);
9892                            <1>       ;outw(VBE_DISPI_IOPORT_DATA, enable);
9893                            <1>
9894                            <1>       ; 23/01/2021
9895 00003F8D B001              <1>       mov    al, 1 ; VBE_DISPI_ENABLED
9896                            <1>       ; ax = 1
9897 00003F8F E8E5FFFFFF        <1>       call   vbe_bfn_rvs_0
9898                            <1>
9899                            <1>       ; from VBE_DISPI_INDEX_BANK
9900                            <1>       ;   to VBE_DISPI_INDEX_Y_OFFSET
9901                            <1>
9902 00003F94 B505              <1>       mov    ch, 5
9903                            <1>       ; 23/01/2021
9904 00003F96 B004              <1>       mov    al, 4  ; VBE_DISPI_INDEX_BANK - 1
9905                            <1>       ; ax = 4
9906                            <1> vbe_bfn_rvs_2:
9907 00003F98 FEC0              <1>       inc    al ; from VBE_DISPI_INDEX_XRES
9908                            <1>                     ; to VBE_DISPI_INDEX_BPP
9909                            <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
9910                            <1>       ;mov   dl, 0CEh
9911 00003F9A 66EF              <1>       out    dx, ax
9912 00003F9C 50                <1>       push   eax
9913                            <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
9914 00003F9D FEC2              <1>       inc    dl ; 1CFh
9915 00003F9F 66AD              <1>       lodsw
9916 00003FA1 66EF              <1>       out    dx, ax
9917 00003FA3 58                <1>       pop    eax
9918 00003FA4 FECA              <1>       dec    dl ; 1CEh
```

```
                        <1>       dec    ch
                        <1>       jnz    short vbe_bfn_rvs_2
                        <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
00003FAB 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>
00003FAF 50             <1>       push   eax
00003FB0 6689D0         <1>       mov    ax, dx
00003FB3 66BACE01       <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
00003FB7 66EF           <1>       out    dx, ax
00003FB9 58             <1>       pop    eax
                        <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
                        <1>       ;mov   dl, 0CFh
00003FBA FEC2           <1>       inc    dl
00003FBC 66EF           <1>       out    dx, ax
00003FBE 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
                        <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ;mov   ax, 03h        ; VBE_DISPI_INDEX_BPP
                        <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
00003FBF 66BA0300       <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
                        <1>       ;call  dispi_set_parms
                        <1>       ;pop   edx
                        <1>       ;retn
00003FC3 EBEA           <1>       jmp    short dispi_set_parms
                        <1>
                        <1> dispi_set_xres:
                        <1>       ; 25/11/2020
                        <1>       ; Input:
                        <1>       ;      ax = X resolution (screen witdh)
                        <1>       ;           (320,640,800,1024,1280,1920)
                        <1>       ;
                        <1>       ; Modified registers: none
                        <1>
                        <1>       ;push  edx
                        <1>       ;push  eax
                        <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
                        <1>       ;mov   ax, 01h        ; VBE_DISPI_INDEX_XRES
                        <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
```

```
10024                                  <1>
10025                                  <1>       ; 25/11/2020
10026                                  <1>       ; Modified registers: edx
10027                                  <1>       ;push  edx
10028 00003FC5 66BA0100                <1>       mov    dx, 01h ; VBE_DISPI_INDEX_XRES
10029                                  <1>       ;call  dispi_set_parms
10030                                  <1>       ;pop   edx
10031                                  <1>       ;retn
10032 00003FC9 EBE4                    <1>       jmp    short dispi_set_parms
10033                                  <1>
10034                                  <1> dispi_set_yres:
10035                                  <1>       ; 25/11/2020
10036                                  <1>       ; Input:
10037                                  <1>       ;      ax = Y resolution (screen height)
10038                                  <1>       ;           (200,400,600,720,768,1080)
10039                                  <1>       ;
10040                                  <1>       ; Modified registers: none
10041                                  <1>
10042                                  <1>       ;push  edx
10043                                  <1>       ;push  eax
10044                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10045                                  <1>       ;mov   ax, 02h         ; VBE_DISPI_INDEX_YRES
10046                                  <1>       ;out   dx, ax
10047                                  <1>       ;pop   eax
10048                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10049                                  <1>       ;;mov  dl, 0CFh
10050                                  <1>       ;inc   dl
10051                                  <1>       ;out   dx, ax
10052                                  <1>       ;pop   edx
10053                                  <1>       ;retn
10054                                  <1>
10055                                  <1>       ; 25/11/2020
10056                                  <1>       ; Modified registers: edx
10057                                  <1>       ;push  edx
10058 00003FCB 66BA0200                <1>       mov    dx, 02h  ; VBE_DISPI_INDEX_YRES
10059                                  <1>       ;call  dispi_set_parms
10060                                  <1>       ;pop   edx
10061                                  <1>       ;retn
10062 00003FCF EBDE                    <1>       jmp    short dispi_set_parms
10063                                  <1>
10064                                  <1> dispi_set_bank:
10065                                  <1>       ; 25/11/2020
10066                                  <1>       ; Input:
10067                                  <1>       ;      ax = video memory bank number
10068                                  <1>       ;
10069                                  <1>       ; Modified registers: none
10070                                  <1>
10071                                  <1>       ;push  edx
10072                                  <1>       ;push  eax
10073                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10074                                  <1>       ;mov   ax, 05h         ; VBE_DISPI_INDEX_BANK
10075                                  <1>       ;out   dx, ax
10076                                  <1>       ;pop   eax
10077                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10078                                  <1>       ;;mov  dl, 0CFh
10079                                  <1>       ;inc   dl
10080                                  <1>       ;out   dx, ax
10081                                  <1>       ;pop   edx
10082                                  <1>       ;retn
10083                                  <1>
10084                                  <1>       ; 25/11/2020
10085                                  <1>       ; Modified registers: edx
10086                                  <1>       ;push  edx
10087 00003FD1 66BA0500                <1>       mov    dx, 05h  ; VBE_DISPI_INDEX_BANK
10088                                  <1>       ;call  dispi_set_parms
10089                                  <1>       ;pop   edx
10090                                  <1>       ;retn
10091 00003FD5 EBD8                    <1>       jmp    short dispi_set_parms
10092                                  <1>
10093                                  <1> dispi_get_enable:
10094                                  <1>       ; 27/11/2020
10095                                  <1>       ; Input:
10096                                  <1>       ;      none
10097                                  <1>       ; Output:
10098                                  <1>       ;      ax = vbe dispi status
10099                                  <1>       ;
10100                                  <1>       ; Modified registers: eax
10101                                  <1>
10102                                  <1>       ;push  edx
10103                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10104                                  <1>       ;mov   ax, 04h         ; VBE_DISPI_INDEX_ENABLE
10105                                  <1>       ;out   dx, ax
10106                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10107                                  <1>       ;;mov  dl, 0CFh
10108                                  <1>       ;inc   dl
10109                                  <1>       ;in    ax, dx
10110                                  <1>       ;pop   edx
10111                                  <1>       ;retn
10112                                  <1>
10113                                  <1>       ; 27/11/2020
10114                                  <1>       ; Modified registers: eax, edx
10115                                  <1>       ;push  edx
10116 00003FD7 66B80400                <1>       mov    ax, 04h ; VBE_DISPI_INDEX_ENABLE
10117                                  <1>       ;call  dispi_get_parms
10118                                  <1>       ;pop   edx
10119                                  <1>       ;retn
10120                                  <1>       ;;jmp  short dispi_get_parms
10121                                  <1>
10122                                  <1> dispi_get_parms:
10123                                  <1>       ; 25/11/2020
10124                                  <1>       ; Input:
10125                                  <1>       ;      ax = vbe dispi register index
10126                                  <1>       ; output:
10127                                  <1>       ;      ax = data
10128                                  <1>       ;
```

```
10129                                  <1>       ; Modified registers: eax, edx
10130                                  <1>
10131 00003FDB 66BACE01                <1>       mov    dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10132 00003FDF 66EF                    <1>       out    dx, ax
10133                                  <1>       ;mov   dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10134                                  <1>       ;mov   dl, 0CFh
10135 00003FE1 FEC2                    <1>       inc    dl
10136 00003FE3 66ED                    <1>       in     ax, dx
10137 00003FE5 C3                      <1>       retn
10138                                  <1>
10139                                  <1> vga_compat_setup:
10140                                  <1>       ; 26/11/2020
10141                                  <1>       ; 25/11/2020
10142                                  <1>       ; VGA compatibility setup
10143                                  <1>       ; (vbe.c, 02/01/2020, vruppert)
10144                                  <1>       ;
10145                                  <1>       ; Input:
10146                                  <1>       ;      none
10147                                  <1>       ;
10148                                  <1>       ; Modified registers: eax, edx
10149                                  <1>
10150                                  <1>       ; 26/11/2020
10151                                  <1>       ;push  eax
10152                                  <1>       ;push  edx
10153                                  <1>
10154                                  <1>       ; set CRT X resolution
10155 00003FE6 66BACE01                <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10156 00003FEA 66B80100                <1>       mov    ax, 01h  ; VBE_DISPI_INDEX_XRES
10157 00003FEE 66EF                    <1>       out    dx, ax
10158                                  <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10159 00003FF0 FEC2                    <1>       inc    dl
10160 00003FF2 66ED                    <1>       in     ax, dx
10161 00003FF4 50                      <1>       push   eax
10162 00003FF5 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10163 00003FF9 66B81100                <1>       mov    ax, 0011h ; Vertical retrace end register
10164 00003FFD 66EF                    <1>       out    dx, ax
10165                                  <1>       ;pop   eax
10166                                  <1>       ;push  eax
10167 00003FFF 8B0424                  <1>       mov    eax, [esp]
10168 00004002 66C1E803                <1>       shr    ax, 3 ; / 8 for pixel to character
10169 00004006 6648                    <1>       dec    ax  ; - 1 (EGA or VGA?)
10170 00004008 88C4                    <1>       mov    ah, al
10171 0000400A B001                    <1>       mov    al, 01h ; Horizontal display end register
10172 0000400C 66EF                    <1>       out    dx, ax
10173 0000400E 58                      <1>       pop    eax
10174                                  <1>
10175 0000400F E8B0000000              <1>       call   vga_set_virt_width
10176                                  <1>
10177                                  <1>       ; set CRT Y resolution
10178 00004014 66BACE01                <1>       mov    dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10179 00004018 66B80200                <1>       mov    ax, 02h  ; VBE_DISPI_INDEX_YRES
10180 0000401C 66EF                    <1>       out    dx, ax
10181                                  <1>       ;mov   dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10182 0000401E FEC2                    <1>       inc    dl
10183 00004020 66ED                    <1>       in     ax, dx
10184 00004022 50                      <1>       push   eax
10185 00004023 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10186 00004027 88C4                    <1>       mov    ah, al
10187 00004029 B012                    <1>       mov    al, 12h ; Vertical display end register
10188 0000402B 66EF                    <1>       out    dx, ax
10189 0000402D 58                      <1>       pop    eax
10190 0000402E B007                    <1>       mov    al, 07h      ; Overflow register
10191 00004030 EE                      <1>       out    dx, al
10192 00004031 6642                    <1>       inc    dx
10193 00004033 EC                      <1>       in     al, dx ; read overflow register
10194 00004034 24BD                    <1>       and    al, 0BDh ; clear VDE 9th and 10th bits
10195 00004036 F6C401                  <1>       test   ah, 01h
10196 00004039 7402                    <1>       jz     short bit8_clear
10197 0000403B 0C02                    <1>       or     al, 02h ; VDE 9th bit (bit 8) in bit 1
10198                                  <1> bit8_clear:
10199 0000403D F6C402                  <1>       test   ah, 02h
10200 00004040 7402                    <1>       jz     short bit9_clear
10201 00004042 0C40                    <1>       or     al, 40h ; VDE 10th bit (bit 9) in bit 6
10202                                  <1> bit9_clear:
10203 00004044 EE                      <1>       out    dx, al
10204                                  <1>
10205                                  <1>       ; other settings
10206 00004045 66BAD403                <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10207 00004049 66B80900                <1>       mov    ax, 0009h ; Maximum scan line register
10208 0000404D 66EF                    <1>       out    dx, ax ; Reset
10209 0000404F B017                    <1>       mov    al, 17h ; Mode control register
10210 00004051 EE                      <1>       out    dx, al
10211                                  <1>       ;mov   dx, 3D5h ; VGAREG_VGA_CRTC_DATA
10212 00004052 FEC2                    <1>       inc    dl
10213 00004054 EC                      <1>       in     al, dx ; Read mode control register
10214 00004055 0C03                    <1>       or     al, 03h ; Set SRS and CMS bits
10215 00004057 EE                      <1>       out    dx, al
10216 00004058 66BADA03                <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10217 0000405C EC                      <1>       in     al, dx  ; clear flip-flop
10218 0000405D 66BAC003                <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10219 00004061 B010                    <1>       mov    al, 10h      ; Mode control register
10220 00004063 EE                      <1>       out    dx, al
10221                                  <1>       ;mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10222 00004064 FEC2                    <1>       inc    dl
10223 00004066 EC                      <1>       in     al, dx
10224 00004067 0C01                    <1>       or     al, 01h ; select graphics mode
10225                                  <1>       ;mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10226 00004069 FECA                    <1>       dec    dl
10227 0000406B EE                      <1>       out    dx, al ; Write to mode control register
10228 0000406C B020                    <1>       mov    al, 20h ; Palette RAM <-> display memory
10229 0000406E EE                      <1>       out    dx, al ; Write to attribute addr register
10230 0000406F 66BACE03                <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
10231 00004073 66B80605                <1>       mov    ax, 0506h ; Misc. register, graph, mm 1
10232 00004077 66EF                    <1>       out    dx, ax
10233 00004079 66BAC403                <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
```

```
10234 0000407D 66B8020F            <1>       mov    ax, 0F02h ; Map mask register, all planes
10235 00004081 66EF                <1>       out    dx, ax
10236                              <1>
10237                              <1>       ; settings for >= 8bpp
10238                              <1>
10239                              <1>       ;mov   dx, 1CEh ; VBE_DISPI_IOPORT_INDEX
10240                              <1>       ;mov   ax, 03h ; VBE_DISPI_INDEX_BPP
10241                              <1>       ;out   dx, ax
10242                              <1>       ;;mov  dx, 1CFh ; VBE_DISPI_IOPORT_DATA
10243                              <1>       ;inc   dl
10244                              <1>       ;in    ax, dx
10245                              <1>       ;cmp   al, 08h  ; < 8 bits per pixel
10246                              <1>       ;jb    short vga_compat_end
10247                              <1>
10248 00004083 66BAD403           <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10249 00004087 B014                <1>       mov    al, 14h  ; Underline location register
10250 00004089 EE                  <1>       out    dx, al
10251                              <1>       ;mov   dx, 3D5h ; VGAREG_VGA_CRTC_DATA
10252 0000408A FEC2                <1>       inc    dl
10253 0000408C EC                  <1>       in     al, dx
10254 0000408D 0C40                <1>       or     al, 40h       ; enable double word mode
10255 0000408F EE                  <1>       out    dx, al
10256 00004090 66BADA03            <1>       mov    dx, 3DAh ; VGAREG_ACTL_RESET
10257 00004094 EC                  <1>       in     al, dx  ; clear flip-flop
10258 00004095 66BAC003            <1>       mov    dx, 3C0h ; VGAREG_ACTL_ADDRESS
10259 00004099 B010                <1>       mov    al, 10h       ; Mode control register
10260 0000409B EE                  <1>       out    dx, al
10261                              <1>       ;mov   dx, 3C1h ; VGAREG_ACTL_READ_DATA
10262 0000409C FEC2                <1>       inc    dl
10263 0000409E EC                  <1>       in     al, dx
10264 0000409F 0C40                <1>       or     al, 40h  ; Pixel clock select is 1
10265                              <1>       ;mov   dx, 3C0h ; VGAREG_ACTL_ADDRESS
10266 000040A1 FECA                <1>       dec    dl
10267 000040A3 EE                  <1>       out    dx, al  ; update mode control reggister
10268 000040A4 B020                <1>       mov    al, 20h       ; select display memory as PAS
10269 000040A6 EE                  <1>       out    dx, al
10270 000040A7 66BAC403            <1>       mov    dx, 3C4h ; VGAREG_SEQU_ADDRESS
10271 000040AB B004                <1>       mov    al, 04h       ; Memory mode register
10272 000040AD EE                  <1>       out    dx, al
10273                              <1>       ;mov   dx, 3C5h ; VGAREG_SEQU_DATA
10274 000040AE FEC2                <1>       inc    dl
10275 000040B0 EC                  <1>       in     al, dx
10276 000040B1 0C08                <1>       or     al, 08h       ; enable chain four
10277 000040B3 EE                  <1>       out    dx, al
10278 000040B4 66BACE03            <1>       mov    dx, 3CEh ; VGAREG_GRDC_ADDRESS
10279 000040B8 B005                <1>       mov    al, 05h       ; Mode register
10280 000040BA EE                  <1>       out    dx, al
10281                              <1>       ;mov   dx, 3CFh ; VGAREG_GRDC_DATA
10282 000040BB FEC2                <1>       inc    dl
10283 000040BD EC                  <1>       in     al, dx
10284 000040BE 249F                <1>       and    al, 9Fh       ; clear shift register
10285 000040C0 0C40                <1>       or     al, 40h  ; set shift register to 2
10286 000040C2 EE                  <1>       out    dx, al
10287                              <1>
10288                              <1> vga_compat_end:
10289                              <1>       ;pop   edx
10290                              <1>       ;pop   eax
10291 000040C3 C3                  <1>       retn
10292                              <1>
10293                              <1> vga_set_virt_width:
10294                              <1>       ; 27/11/2020
10295                              <1>       ; 25/11/2020
10296                              <1>       ; (vbe.c, 02/01/2020, vruppert)
10297                              <1>       ;
10298                              <1>       ; Input:
10299                              <1>       ;      AX = resolution (screen width)
10300                              <1>       ;
10301                              <1>       ; Modified registers: eax, edx
10302                              <1>
10303                              <1>       ;;push ebx
10304                              <1>       ;push  edx
10305                              <1>       ;push  eax
10306                              <1>       ;mov   ebx, eax
10307                              <1>       ;call  dispi_get_bpp ; bits per pixel
10308                              <1>       ;cmp   al, 4
10309                              <1>       ;ja    short set_width_svga  ; 8, 16, 24, 32
10310                              <1>       ;shr   bx, 1
10311                              <1> ;set_width_svga:
10312                              <1>       ;shr   bx, 3
10313                              <1>       ;mov   eax, [esp]
10314 000040C4 66C1E803           <1>       shr    ax, 3  ; / 8, bytes per row
10315 000040C8 66BAD403            <1>       mov    dx, 3D4h ; VGAREG_VGA_CRTC_ADDRESS
10316                              <1>       ;mov   ah, bl ;
10317 000040CC 88C4                <1>       mov    ah, al ; width in bytes
10318 000040CE B013                <1>       mov    al, 13h       ; offset register
10319 000040D0 66EF                <1>       out    dx, ax ; index (3D4h) and data (3D5h)
10320                              <1>       ;pop   eax
10321                              <1>       ;pop   edx
10322                              <1>       ;;pop  ebx
10323 000040D2 C3                  <1>       retn
10324                              <1>
10325                              <1> ; 24/11/2020
10326                              <1>
10327                              <1> struc bmi ; BOCHS/PLEX86 MODE INFO structure/table
10328 00000000 <res 00000002>      <1>  .mode:       resw 1
10329 00000002 <res 00000002>      <1>  .width:  resw 1
10330 00000004 <res 00000002>      <1>  .height: resw 1
10331 00000006 <res 00000002>      <1>  .depth:  resw 1
10332                              <1>  .size:
10333                              <1> endstruc
10334                              <1>
10335                              <1> ; 24/11/2020
10336                              <1> struc MODEINFO
10337 00000000 <res 00000002>      <1>  .mode:                   resw 1  ; 1XXh
10338 00000002 <res 00000002>      <1>  .ModeAttributes:  resw 1
```

```
10339 00000004 <res 00000001>      <1>  .WinAAttributes:  resb 1
10340 00000005 <res 00000001>      <1>  .WinBAttributes:  resb 1 ; = 0
10341 00000006 <res 00000002>      <1>  .WinGranularity:  resw 1
10342 00000008 <res 00000002>      <1>  .WinSize:         resw 1
10343 0000000A <res 00000002>      <1>  .WinASegment:            resw 1
10344 0000000C <res 00000002>      <1>  .WinBSegment:            resw 1 ; = 0
10345 0000000E <res 00000004>      <1>  .WinFuncPtr:             resd 1 ; = 0
10346 00000012 <res 00000002>      <1>  .BytesPerScanLine:       resw 1
10347 00000014 <res 00000002>      <1>  .XResolution:            resw 1
10348 00000016 <res 00000002>      <1>  .YResolution:            resw 1
10349 00000018 <res 00000001>      <1>  .XCharSize:       resb 1
10350 00000019 <res 00000001>      <1>  .YCharSize:       resb 1
10351 0000001A <res 00000001>      <1>  .NumberOfPlanes:  resb 1
10352 0000001B <res 00000001>      <1>  .BitsPerPixel:           resb 1
10353 0000001C <res 00000001>      <1>  .NumberOfBanks:   resb 1
10354 0000001D <res 00000001>      <1>  .MemoryModel:            resb 1
10355 0000001E <res 00000001>      <1>  .BankSize:        resb 1 ; = 0
10356 0000001F <res 00000001>      <1>  .NumberOfImagePages:     resb 1
10357 00000020 <res 00000001>      <1>  .Reserved_page:   resb 1 ; = 0
10358 00000021 <res 00000001>      <1>  .RedMaskSize:            resb 1
10359 00000022 <res 00000001>      <1>  .RedFieldPosition:       resb 1
10360 00000023 <res 00000001>      <1>  .GreenMaskSize:   resb 1
10361 00000024 <res 00000001>      <1>  .GreenFieldPosition:     resb 1
10362 00000025 <res 00000001>      <1>  .BlueMaskSize:           resb 1
10363 00000026 <res 00000001>      <1>  .BlueFieldPosition:      resb 1
10364 00000027 <res 00000001>      <1>  .RsvdMaskSize:           resb 1
10365 00000028 <res 00000001>      <1>  .RsvdFieldPosition:      resb 1
10366 00000029 <res 00000001>      <1>  .DirectColorModeInfo:    resb 1
10367 0000002A <res 00000004>      <1>  .PhysBasePtr:            resd 1
10368 0000002E <res 00000004>      <1>  .OffScreenMemOffset:     resd 1 ; = 0
10369 00000032 <res 00000002>      <1>  .OffScreenMemSize:       resw 1 ; = 0
10370 00000034 <res 00000002>      <1>  .LinBytesPerScanLine:    resw 1
10371 00000036 <res 00000001>      <1>  .BnkNumberOfPages:       resb 1
10372 00000037 <res 00000001>      <1>  .LinNumberOfPages:       resb 1
10373 00000038 <res 00000001>      <1>  .LinRedMaskSize:  resb 1
10374 00000039 <res 00000001>      <1>  .LinRedFieldPosition1: resb 1
10375 0000003A <res 00000001>      <1>  .LinGreenMaskSize1:      resb 1
10376 0000003B <res 00000001>      <1>  .LinGreenFieldPosition:resb 1
10377 0000003C <res 00000001>      <1>  .LinBlueMaskSize:        resb 1
10378 0000003D <res 00000001>      <1>  .LinBlueFieldPosition:   resb 1
10379 0000003E <res 00000001>      <1>  .LinRsvdMaskSize:        resb 1
10380 0000003F <res 00000001>      <1>  .LinRsvdFieldPosition:   resb 1
10381 00000040 <res 00000004>      <1>  .MaxPixelClock:   resd 1 ; = 0
10382                              <1> .size:
10383                              <1> endstruc
10384                              <1>
10385                              <1> ; 10/12/2020
10386                              <1> struc LFBINFO
10387 00000000 <res 00000002>      <1>  .mode:                   resw 1  ; 1XXh
10388 00000002 <res 00000004>      <1>  .LFB_addr:        resd 1
10389 00000006 <res 00000004>      <1>  .LFB_size:        resd 1
10390 0000000A <res 00000002>      <1>  .X_res:           resw 1
10391 0000000C <res 00000002>      <1>  .Y_res:           resw 1
10392 0000000E <res 00000001>      <1>  .bpp:                    resb 1
10393 0000000F <res 00000001>      <1>  .reserved:        resb 1
10394                              <1> .size:      ; 16 bytes
10395                              <1> endstruc
10396                              <1>
10397                              <1> set_mode_info_list:
10398                              <1>       ; 14/12/2020
10399                              <1>       ; 11/12/2020
10400                              <1>       ; 24/11/2020
10401                              <1>       ; (vbetables-gen.c)
10402                              <1>       ; Input:
10403                              <1>       ;     BX = VBE mode (including bochs special modes)
10404                              <1>       ; Output:
10405                              <1>       ;     ;;EAX = MODE_INFO_LIST address
10406                              <1>       ;     EAX = 0 ; 11/12/2020
10407                              <1>       ;     ESI = MODE_INFO_LIST address ; 11/12/2020
10408                              <1>       ;     (if mode is not found, ESI = 0)
10409                              <1>       ;
10410                              <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
10411                              <1>
10412 000040D3 BE[66730000]       <1>       mov   esi, b_vbe_modes ; bochs mode info base table
10413 000040D8 BF[3A120300]       <1>       mov   edi, MODE_INFO_LIST ; mode info list (4F01h)
10414                              <1> sml_0:
10415 000040DD 66AD               <1>       lodsw
10416 000040DF 6639D8             <1>       cmp   ax, bx ; is mode number same ?
10417 000040E2 7410               <1>       je    short sml_1 ; yes
10418 000040E4 AD                 <1>       lodsd
10419 000040E5 66AD               <1>       lodsw
10420 000040E7 81FE[26740000]     <1>       cmp   esi, end_of_b_vbe_modes
10421 000040ED 72EE               <1>       jb    short sml_0
10422                              <1>       ; not found
10423 000040EF 31C0               <1>       xor   eax, eax ; 0
10424                              <1>       ; 11/12/2020
10425 000040F1 31F6               <1>       xor   esi, esi
10426 000040F3 C3                 <1>       retn
10427                              <1> sml_1:
10428 000040F4 66AB               <1>       stosw  ; mode
10429 000040F6 AD                 <1>       lodsd  ; width, height
10430                              <1>       ; 14/12/2020
10431 000040F7 89C1               <1>       mov   ecx, eax
10432 000040F9 50                 <1>       push  eax ; ***
10433 000040FA 29C0               <1>       sub   eax, eax ; clear high word of eax
10434 000040FC 66AD               <1>       lodsw  ; depth
10435 000040FE 50                 <1>       push  eax ; **
10436                              <1>
10437                              <1>       ;add  al, 7 ; only for 15 bit colors (not used here)
10438 000040FF C0E803             <1>       shr   al, 3 ; / 8
10439                              <1>       ; 14/12/2020
10440 00004102 66F7E1             <1>       mul   cx  ; pitch = width * ((depth+7)/8)
10441                              <1>             ; ax = pitch
10442 00004105 50                 <1>       push  eax ; * ; high word of eax = 0
10443 00004106 C1E910             <1>       shr   ecx, 16
```

```
10444                                  <1>       ;mul   cx
10445                                  <1>       ;mov   cx, ax
10446 00004109 31D2                    <1>       xor    edx, edx  ; clear high word of edx
10447 0000410B F7E1                    <1>       mul    ecx ; height * pitch
10448 0000410D 89C1                    <1>       mov    ecx, eax
10449 0000410F B800000001              <1>       mov    eax, VBE_DISPI_TOTAL_VIDEO_MEMORY_MB * 1024 * 1024
10450 00004114 F7F1                    <1>       div    ecx
10451                                  <1>              ; eax = pages = vram_size / (height*pitch)
10452                                  <1>
10453                                  <1>       ;mov   cx, ax
10454 00004116 89C1                    <1>       mov    ecx, eax ; pages
10455                                  <1>
10456 00004118 66B89B00                <1>       mov    ax, MODE_ATTRIBUTES
10457 0000411C 66AB                    <1>       stosw  ; ModeAttributes
10458 0000411E B007                    <1>       mov    al, WINA_ATTRIBUTES
10459 00004120 AA                      <1>       stosb  ; WinAAttributes
10460 00004121 30C0                    <1>       xor    al, al ; WinBAttributes = 0
10461 00004123 AA                      <1>       stosb
10462 00004124 66B84000                <1>       mov    ax, VBE_DISPI_BANK_SIZE_KB
10463 00004128 66AB                    <1>       stosw  ; WinGranularity
10464 0000412A 66AB                    <1>       stosw  ; WinSize
10465 0000412C 66B800A0                <1>       mov    ax, VGAMEM_GRAPH
10466 00004130 66AB                    <1>       stosw  ; WinASegment
10467 00004132 29C0                    <1>       sub    eax, eax
10468 00004134 66AB                    <1>       stosw  ; WinBSegment = 0
10469 00004136 AB                      <1>       stosd  ; WinFuncPtr = 0
10470                                  <1>
10471 00004137 58                      <1>       pop    eax ; * ; pitch
10472 00004138 89C3                    <1>       mov    ebx, eax  ; high word of ebx = 0 ; 14/12/2020
10473 0000413A 66AB                    <1>       stosw  ; BytesPerScanLine
10474                                  <1>
10475 0000413C 5A                      <1>       pop    edx ; ** ; depth (bits per pixel)
10476 0000413D 58                      <1>       pop    eax ; *** width, height
10477                                  <1>
10478                                  <1>       ; // Mandatory information for VBE 1.2 and above
10479                                  <1>
10480 0000413E 66AB                    <1>       stosw  ; XResolution (width)
10481 00004140 C1E810                  <1>       shr    eax, 16
10482 00004143 50                      <1>       push   eax ; **** height
10483 00004144 66AB                    <1>       stosw  ; YResolution (height)
10484 00004146 B008                    <1>       mov    al, 8
10485 00004148 AA                      <1>       stosb  ; XCharSize  ; char width
10486 00004149 B010                    <1>       mov    al, 16
10487 0000414B AA                      <1>       stosb  ; YCharSize  ; char height
10488 0000414C B001                    <1>       mov    al, 1
10489 0000414E AA                      <1>       stosb  ; NumberOfPlanes
10490                                  <1>       ;movzx eax, dl
10491 0000414F 88D0                    <1>       mov    al, dl ; eax <= 32
10492 00004151 AA                      <1>       stosb  ; BitsPerPixel
10493                                  <1>       ; Number of banks = (height * pitch + 65535) / 65536
10494 00004152 58                      <1>       pop    eax ; **** ; height
10495                                  <1>       ; 14/12/2020
10496 00004153 52                      <1>       push   edx ; ***** ; depth ; edx <= 32
10497 00004154 F7E3                    <1>       mul    ebx  ; pitch (ebx) * height (eax)
10498                                  <1>       ;mov   edx, [esp] ; *****
10499                                  <1>       ;mov   dl, [esp] ; *****
10500 00004156 05FFFF0000              <1>       add    eax, 65535
10501 0000415B C1E810                  <1>       shr    eax, 16 ; / 65536 ; <= 127 ; 14/12/2020
10502 0000415E AA                      <1>       stosb  ; NumberOfBanks
10503                                  <1>       ; 14/12/2020
10504                                  <1>       ;cmp   dl, 8 ; 8 bits per pixel
10505 0000415F 803C2408                <1>       cmp    byte [esp], 8
10506 00004163 7704                    <1>       ja     short sml_2
10507 00004165 B004                    <1>       mov    al, VBE_MEMORYMODEL_PACKED_PIXEL
10508 00004167 EB02                    <1>       jmp    short sml_3
10509                                  <1> sml_2:
10510                                  <1>       ; 16, 24, 32 bts per pixel
10511 00004169 B006                    <1>       mov    al, VBE_MEMORYMODEL_DIRECT_COLOR
10512                                  <1> sml_3:
10513 0000416B AA                      <1>       stosb
10514 0000416C 30C0                    <1>       xor    al, al ; 0
10515 0000416E AA                      <1>       stosb  ; BankSize = 0
10516 0000416F 49                      <1>       dec    ecx ; pages - 1
10517                                  <1>       ; NumberOfImagePages = 262 for 320x200x8 mode
10518                                  <1>       ;;mov  ax, 255
10519                                  <1>       ; 14/12/2020
10520                                  <1>       ;mov   al, 255
10521 00004170 FEC8                    <1>       dec    al ; 255
10522 00004172 39C1                    <1>       cmp    ecx, eax ; ecx <= 261, eax = 255
10523                                  <1>       ;cmp   cx, ax
10524 00004174 7302                    <1>       jnb    short sml_4
10525 00004176 88C8                    <1>       mov    al, cl
10526                                  <1> sml_4:
10527 00004178 AA                      <1>       stosb  ; NumberOfImagePages (1 byte)
10528 00004179 28C0                    <1>       sub    al, al
10529 0000417B AA                      <1>       stosb  ; Reserved_page = 0
10530 0000417C 58                      <1>       pop    eax ; ***** ; depth
10531 0000417D 88C1                    <1>       mov    cl, al
10532                                  <1>       ; eax < = 32
10533 0000417F 2C08                    <1>       sub    al, 8 ; 8->0, 16->8, 24->16, 32->24
10534 00004181 BE[26740000]            <1>       mov    esi, direct_color_fields
10535 00004186 01C6                    <1>       add    esi, eax
10536 00004188 56                      <1>       push   esi ; ******
10537 00004189 AD                      <1>       lodsd  ; RedMaskSize (AL), RedFieldPosition (AH)
10538                                  <1>              ; GreenMaskSize (16), GreenFieldPosition (24)
10539 0000418A AB                      <1>       stosd
10540 0000418B AD                      <1>       lodsd  ; BlueMaskSize (AL), BlueFieldPosition (AH)
10541                                  <1>              ; RsvdMaskSize (16), RsvdFieldPosition (24)
10542 0000418C AB                      <1>       stosd
10543 0000418D 5E                      <1>       pop    esi ; ******
10544                                  <1>
10545 0000418E 30C0                    <1>       xor    al, al ; 0
10546 00004190 80F920                  <1>       cmp    cl, 32
10547 00004193 7202                    <1>       jb     short sml_5
10548 00004195 B002                    <1>       mov    al, VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
```

```
10549                                  <1> sml_5:
10550 00004197 AA                      <1>       stosb  ; DirectColorModeInfo
10551                                  <1>
10552                                  <1>       ; // Mandatory information for VBE 2.0 and above
10553                                  <1>
10554 00004198 B8000000E0              <1>       mov    eax, VBE_DISPI_LFB_PHYSICAL_ADDRESS
10555 0000419D AB                      <1>       stosd  ; PhysBasePtr
10556 0000419E 29C0                    <1>       sub    eax, eax
10557 000041A0 AB                      <1>       stosd  ; OffScreenMemOffset = 0
10558 000041A1 66AB                    <1>       stosw  ; OffScreenMemSize = 0
10559                                  <1>
10560                                  <1>       ;// Mandatory information for VBE 3.0 and above
10561                                  <1>
10562                                  <1>       ; ebx = pitch
10563 000041A3 6689D8                 <1>       mov    ax, bx
10564                                  <1>       ;stosw
10565                                  <1>
10566                                  <1>       ;xor   al, al
10567                                  <1>       ;stosb ; BnkNumberOfPages = 0
10568                                  <1>       ;stosb ; LinNumberOfPages = 0
10569                                  <1>
10570 000041A6 AB                      <1>       stosd  ; pitch (word), 0 (byte), 0 (byte)
10571                                  <1>
10572 000041A7 AD                      <1>       lodsd  ; LinRedMaskSize (AL), LinRedFieldPosition (AH)
10573                                  <1>              ; LinGreenMaskSize (16), LinGreenFieldPosition (24)
10574 000041A8 AB                      <1>       stosd
10575 000041A9 AD                      <1>       lodsd  ; LinBlueMaskSize (AL), LinBlueFieldPosition (AH)
10576                                  <1>              ; LinRsvdMaskSize (16), LinRsvdFieldPosition (24)
10577 000041AA AB                      <1>       stosd
10578                                  <1>
10579 000041AB 29C0                    <1>       sub    eax, eax
10580 000041AD AB                      <1>       stosd  ; MaxPixelClock = 0
10581                                  <1>
10582                                  <1>       ;mov   eax, MODE_INFO_LIST
10583                                  <1>       ; 11/12/2020
10584 000041AE BE[3A120300]            <1>       mov    esi, MODE_INFO_LIST
10585                                  <1>
10586 000041B3 C3                      <1>       retn
10587                                  <1>
10588                                  <1> ; end of set_mode_info_list ; edi = set_mode_info_list + 68
10589                                  <1>
10590                                  <1> pci_get_lfb_addr:
10591                                  <1>       ; 11/12/2020
10592                                  <1>       ; Get linear frame buffer base from PCI
10593                                  <1>       ; (vgabios.c, 02/01/2020, vruppert)
10594                                  <1>       ;
10595                                  <1>       ; Input:
10596                                  <1>       ;      ax = PCI device vendor id
10597                                  <1>       ; Output:
10598                                  <1>       ;      ax  = LFB address (high 16 bit) (zf=0)
10599                                  <1>       ;      eax = 0 -> not found (error) (zf=1)
10600                                  <1>       ;
10601                                  <1>       ; Modified registers: eax
10602                                  <1>
10603 000041B4 53                      <1>       push   ebx
10604 000041B5 51                      <1>       push   ecx
10605 000041B6 52                      <1>       push   edx
10606                                  <1>       ;
10607 000041B7 89C3                    <1>       mov    ebx, eax
10608 000041B9 31C9                    <1>       xor    ecx, ecx
10609 000041BB 28D2                    <1>       sub    dl, dl ; mov dl, 0
10610 000041BD E842000000              <1>       call   pci_read_reg
10611 000041C2 6683F8FF                <1>       cmp    ax, 0FFFFh
10612 000041C6 7417                    <1>       je     short pci_get_lfb_addr_fail
10613                                  <1> pci_get_lfb_addr_next_dev:
10614 000041C8 28D2                    <1>       sub    dl, dl ; mov dl, 0
10615 000041CA E835000000              <1>       call   pci_read_reg
10616 000041CF 6639D8                  <1>       cmp    ax, bx ; check vendor
10617 000041D2 740F                    <1>       je     short pci_get_lfb_addr_found
10618 000041D4 6683C108                <1>       add    cx, 08h
10619 000041D8 6681F90002              <1>       cmp    cx, 200h ; search bus 0 and 1
10620 000041DD 72E9                    <1>       jb     short pci_get_lfb_addr_next_dev
10621                                  <1> pci_get_lfb_addr_fail:
10622 000041DF 31C0                    <1>       xor    eax, eax ; no LFB
10623                                  <1>       ; zf = 1
10624 000041E1 EB1D                    <1>       jmp    short pci_get_lfb_addr_return
10625                                  <1> pci_get_lfb_addr_found:
10626 000041E3 B210                    <1>       mov    dl, 10h      ; I/O space 0
10627 000041E5 E81A000000              <1>       call   pci_read_reg
10628 000041EA 66A9F1FF                <1>       test   ax, 0FFF1h
10629 000041EE 740D                    <1>       jz     short pci_get_lfb_addr_success
10630 000041F0 B214                    <1>       mov    dl, 14h ; I/O space 1
10631 000041F2 E80D000000              <1>       call   pci_read_reg
10632 000041F7 66A9F1FF                <1>       test   ax, 0FFF1h
10633 000041FB 75E2                    <1>       jnz    short pci_get_lfb_addr_fail
10634                                  <1> pci_get_lfb_addr_success:
10635 000041FD C1E810                  <1>       shr    eax, 16 ; LFB address (hw)
10636                                  <1>       ; zf = 0
10637                                  <1> pci_get_lfb_addr_return:
10638 00004200 5A                      <1>       pop    edx
10639 00004201 59                      <1>       pop    ecx
10640 00004202 5B                      <1>       pop    ebx
10641 00004203 C3                      <1>       retn
10642                                  <1>
10643                                  <1> pci_read_reg:
10644                                  <1>       ; 11/12/2020
10645                                  <1>       ; Read PCI register
10646                                  <1>       ; (vgabios.c, 02/01/2020, vruppert)
10647                                  <1>       ;
10648                                  <1>       ; Input:
10649                                  <1>       ;      cx = device/function
10650                                  <1>       ;      dl = register
10651                                  <1>       ; Output:
10652                                  <1>       ;      eax = value
10653                                  <1>       ;
```

```
10654                                  <1>       ; Modified registers: eax, edx
10655                                  <1>
10656 00004204 B800008000              <1>       mov    eax, 00800000h
10657 00004209 6689C8                  <1>       mov    ax, cx
10658 0000420C C1E008                  <1>       shl    eax, 8
10659 0000420F 88D0                    <1>       mov    al, dl
10660 00004211 66BAF80C                <1>       mov    dx, 0CF8h
10661 00004215 EF                      <1>       out    dx, eax
10662 00004216 80C204                  <1>       add    dl, 4 ; mov dx, 0CFCh
10663 00004219 ED                      <1>       in     eax, dx
10664 0000421A C3                      <1>       retn
10665                                  <1>
10666                                  <1> %endif
10667                                  <1>
10668                                  <1> ; -----------
10669                                  <1>
10670                                  <1> %if 0
10671                                  <1>
10672                                  <1> mode_info_find_mode:
10673                                  <1>       ; 25/11/2020
10674                                  <1>       ; Input:
10675                                  <1>       ;      bx = VESA VBE2 video mode (+ bochs extensions)
10676                                  <1>       ; Output:
10677                                  <1>       ;      esi = mode info address (for BX input)
10678                                  <1>       ;      esi = 0 -> not found
10679                                  <1>       ;
10680                                  <1>       ; Modified registers: eax, esi
10681                                  <1>
10682                                  <1>       xor    eax, eax
10683                                  <1>       mov    esi, MODE_INFO_LIST
10684                                  <1> mifm_1:
10685                                  <1>       mov    ax, [esi]
10686                                  <1>       cmp    ax, bx
10687                                  <1>       je     short mifm_2
10688                                  <1>       add    esi, MODEINFO.size ; add esi, 68
10689                                  <1>       cmp    esi, VBE_VESA_MODE_END_OF_LIST
10690                                  <1>       jb     short mifm_1
10691                                  <1>       ; not found
10692                                  <1>       sub    esi, esi ; 0
10693                                  <1> mifm_2
10694                                  <1>       retn
10695                                  <1>
10696                                  <1> dispi_get_bpp:
10697                                  <1>       ; 28/11/2020
10698                                  <1>       ; Input:
10699                                  <1>       ;      none
10700                                  <1>       ; Output:
10701                                  <1>       ;      al = Bits per pixel
10702                                  <1>       ;           (8,16,24,32)
10703                                  <1>       ;      ah = Bytes per pixel
10704                                  <1>       ;           (1,2,3,4)
10705                                  <1>       ;
10706                                  <1>       ; Modified registers: none
10707                                  <1>
10708                                  <1>       ;push  edx
10709                                  <1>       ;mov   dx, 01CEh ; VBE_DISPI_IOPORT_INDEX
10710                                  <1>       ;mov   ax, 03h        ; VBE_DISPI_INDEX_BPP
10711                                  <1>       ;out   dx, ax
10712                                  <1>       ;;mov  dx, 01CFh ; VBE_DISPI_IOPORT_DATA
10713                                  <1>       ;;mov  dl, 0CFh
10714                                  <1>       ;inc   dl
10715                                  <1>       ;in    ax, dx
10716                                  <1>       ;mov   ah, al
10717                                  <1>       ;shr   ah, 3 ; / 8
10718                                  <1>       ;;test al, 7 ; 15 bit graphics mode
10719                                  <1>       ;;jz   short get_bpp_noinc
10720                                  <1>       ;;inc  ah
10721                                  <1> ;;get_bpp_noinc:
10722                                  <1>       ;pop   edx
10723                                  <1>       ;retn
10724                                  <1>
10725                                  <1>       ; 28/11/2020
10726                                  <1>       ; Modified registers: edx
10727                                  <1>       ;push  edx
10728                                  <1>       mov    dx, 03h ; VBE_DISPI_INDEX_BPP
10729                                  <1>       call   dispi_get_parms
10730                                  <1>       ;pop   edx
10731                                  <1>       ;retn
10732                                  <1>       mov    ah, al
10733                                  <1>       shr    ah, 3 ; / 8
10734                                  <1>       ;test  al, 7 ; 15 bit graphips mode
10735                                  <1>       ;jz    short get_bpp_noinc
10736                                  <1>       ;inc   ah
10737                                  <1> ;get_bpp_noinc:
10738                                  <1>       ;pop   edx
10739                                  <1>       retn
10740                                  <1>
10741                                  <1> restore_vesa_video_state:
10742                                  <1>       ; 14/01/2021
10743                                  <1>       ; 06/12/2020
10744                                  <1>       ; Input:
10745                                  <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
10746                                  <1>       ; Output:
10747                                  <1>       ;      AX = 004Fh (successed)
10748                                  <1>       ;
10749                                  <1>       ;      eax = 0 -> buffer size problem
10750                                  <1>       ;      eax > 0 and ax <> 004Fh -> failed
10751                                  <1>       ;
10752                                  <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
10753                                  <1>
10754                                  <1>       ;movzx ecx, word [vbe3stbufsize]
10755                                  <1>       ;cmp   cx, 32 ; 32 * 64 bytes
10756                                  <1>       ;ja    short r_v_b_s_fail
10757                                  <1>
10758                                  <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
```

```
10759                                <1>       shl    cx, 4 ; dword count for movsd
10760                                <1>       mov    edi, VBE3SAVERESTOREBLOCK ; destination
10761                                <1>                                ; (vbe3 pmi buff)
10762                                <1>       mov    esi, VBE3VIDEOSTATE ; source (kernel buff)
10763                                <1>       rep    movsd
10764                                <1>
10765                                <1>       mov    ax, 4F04h
10766                                <1>       mov    dl, 02h ; restore
10767                                <1>       ;mov   cx, 0Fh
10768                                <1>       mov    cl, 0Fh
10769                                <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
10770                                <1>       jmp    short int10h_32bit_pmi
10771                                <1>
10772                                <1> ;s_v_b_s_fail:
10773                                <1> ;r_v_b_s_fail:
10774                                <1> ;     xor    eax, eax
10775                                <1> ;     retn
10776                                <1>
10777                                <1> save_vesa_video_state:
10778                                <1>       ; 14/01/2021
10779                                <1>       ; 06/12/2020
10780                                <1>       ; Input:
10781                                <1>       ;      [vbe3stbufsize] <= 32 ; <= 32*64 bytes
10782                                <1>       ; Output:
10783                                <1>       ;      AX = 004Fh (successed)
10784                                <1>       ;
10785                                <1>       ;      eax = 0 -> buffer size problem
10786                                <1>       ;      eax > 0 and ax <> 004Fh -> failed
10787                                <1>       ;
10788                                <1>       ; Modified regs: eax, ebx, ecx, edx, esi, edi
10789                                <1>
10790                                <1>       ;cmp   word [vbe33stbufsize], 32
10791                                <1>       ;                  ; 32 * 64 bytes
10792                                <1>       ;ja    short s_v_b_s_fail
10793                                <1>
10794                                <1>       mov    ax, 4F04h
10795                                <1>       mov    dl, 01h ; save
10796                                <1>       ;mov   cx, 0Fh
10797                                <1>       mov    cl, 0Fh
10798                                <1>       xor    ebx, ebx ; points to VBE3SAVERESTOREBLOCK
10799                                <1>
10800                                <1>       call   int10h_32bit_pmi
10801                                <1>
10802                                <1>       movzx  ecx, byte [vbe3stbufsize]; <=32
10803                                <1>       shl    cx, 4 ; dword count for movsd
10804                                <1>       mov    esi, VBE3SAVERESTOREBLOCK ; destination
10805                                <1>                                ; (vbe3 pmi buff)
10806                                <1>       mov    edi, VBE3VIDEOSTATE ; source (kernel buff)
10807                                <1>       rep    movsd
10808                                <1>       retn
10809                                <1>
10810                                <1> dispi_set_bank_farcall:
10811                                <1>       ; 11/12/2020
10812                                <1>       ; (This may be 'sysvideo' function, later)
10813                                <1>       ;
10814                                <1>       ; Input:
10815                                <1>       ;      bx = 0000h, set bank number
10816                                <1>       ;         = 0100h, get bank number
10817                                <1>       ;      dx = bank number (if bx = 0)
10818                                <1>       ; Output:
10819                                <1>       ;      dx = bank number
10820                                <1>
10821                                <1>       cmp    bx, 0100h
10822                                <1>       je     short dispi_set_bank_farcall_get
10823                                <1>       or     bx, bx
10824                                <1>       jnz    dispi_set_bank_farcall_error
10825                                <1>       mov    ax, dx
10826                                <1>       push   dx
10827                                <1>       push   ax
10828                                <1>       mov    ax, VBE_DISPI_INDEX_BANK
10829                                <1>       mov    dx, VBE_DISPI_IOPORT_INDEX
10830                                <1>       out    dx, ax
10831                                <1>       pop    ax
10832                                <1>       mov    dx, VBE_DISPI_IOPORT_DATA
10833                                <1>       out    dx, ax
10834                                <1>       in     ax, dx
10835                                <1>       pop    dx
10836                                <1>       cmp    dx, ax
10837                                <1>       jne    short dispi_set_bank_farcall_error
10838                                <1>       mov    ax, 004Fh
10839                                <1>       retn   ; retf for real mode far call
10840                                <1> dispi_set_bank_farcall_get:
10841                                <1>       mov    ax, VBE_DISPI_INDEX_BANK
10842                                <1>       mov    dx, VBE_DISPI_IOPORT_INDEX
10843                                <1>       out    dx, ax
10844                                <1>       mov    dx, VBE_DISPI_IOPORT_DATA
10845                                <1>       in     ax, dx
10846                                <1>       mov    dx, ax
10847                                <1>       retn   ; retf for real mode far call
10848                                <1> dispi_set_bank_farcall_error:
10849                                <1>       mov    ax, 014Fh
10850                                <1>       retn   ; retf for real mode far call
10851                                <1>
10852                                <1> %endif
10853                                <1>
10854                                <1> ; % include 'vidata.s' ; VIDEO DATA
10855                                <1>
10856                                <1> ; /// End Of VIDEO FUNCTIONS ///
 2663
 2664                                    setup_rtc_int:
 2665                                    ; source: http://wiki.osdev.org/RTC
 2666 0000421B FA                         cli          ; disable interrupts
 2667                                     ; default int frequency is 1024 Hz (Lower 4 bits of register A is 0110b or 6)
 2668                                     ; in order to change this ...
 2669                                     ; frequency  = 32768 >> (rate-1) --> 32768 >> 5 = 1024
```

```
2670                                    ; (rate must be above 2 and not over 15)
2671                                    ; new rate = 15 --> 32768 >> (15-1) = 2 Hz
2672 0000421C B08A                      mov   al, 8Ah
2673 0000421E E670                      out   70h, al ; set index to register A, disable NMI
2674 00004220 90                        nop
2675 00004221 E471                      in    al, 71h ; get initial value of register A
2676 00004223 88C4                      mov   ah, al
2677 00004225 80E4F0                    and   ah, 0F0h
2678 00004228 B08A                      mov   al, 8Ah
2679 0000422A E670                      out   70h, al ; reset index to register A
2680 0000422C 88E0                      mov   al, ah
2681 0000422E 0C0F                      or    al, 0Fh      ; new rate (0Fh -> 15)
2682 00004230 E671                      out   71h, al ; write only our rate to A. Note, rate is the bottom 4 bits.
2683                                    ; enable RTC interrupt
2684 00004232 B08B                      mov   al, 8Bh ;
2685 00004234 E670                      out   70h, al ; select register B and disable NMI
2686 00004236 90                        nop
2687 00004237 E471                      in    al, 71h ; read the current value of register B
2688 00004239 88C4                      mov   ah, al  ;
2689 0000423B B08B                      mov   al, 8Bh ;
2690 0000423D E670                      out   70h, al ; set the index again (a read will reset the index to register B)
2691 0000423F 88E0                      mov   al, ah  ;
2692 00004241 0C40                      or    al, 40h ;
2693 00004243 E671                      out   71h, al ; write the previous value ORed with 0x40. This turns on bit 6 of register B
2694 00004245 FB                        sti
2695 00004246 C3                        retn
2696
2697                                ; Write memory information
2698                                ; 29/01/2016
2699                                ; 06/11/2014
2700                                ; 14/08/2015
2701                                memory_info:
2702 00004247 A1[1C8A0100]                mov   eax, [memory_size] ; in pages
2703 0000424C 50                          push  eax
2704 0000424D C1E00C                      shl   eax, 12               ; in bytes
2705 00004250 BB0A000000                  mov   ebx, 10
2706 00004255 89D9                        mov   ecx, ebx        ; 10
2707 00004257 BE[9D4C0100]                mov   esi, mem_total_b_str
2708 0000425C E8D7000000                  call  bintdstr
2709 00004261 58                          pop   eax
2710 00004262 B107                        mov   cl, 7
2711 00004264 BE[C14C0100]                mov   esi, mem_total_p_str
2712 00004269 E8CA000000                  call  bintdstr
2713                                      ; 14/08/2015
2714 0000426E E8E2000000                  call  calc_free_mem
2715                                      ; edx = calculated free pages
2716                                      ; ecx = 0
2717 00004273 A1[208A0100]                mov   eax, [free_pages]
2718 00004278 39D0                        cmp   eax, edx ; calculated free mem value
2719                                            ; and initial free mem value are same or not?
2720 0000427A 751D                        jne   short pmim ; print mem info with '?' if not
2721 0000427C 52                          push  edx ; free memory in pages
2722                                      ;mov  eax, edx
2723 0000427D C1E00C                      shl   eax, 12 ; convert page count
2724                                            ; to byte count
2725 00004280 B10A                        mov   cl, 10
2726 00004282 BE[E14C0100]                mov   esi, free_mem_b_str
2727 00004287 E8AC000000                  call  bintdstr
2728 0000428C 58                          pop   eax
2729 0000428D B107                        mov   cl, 7
2730 0000428F BE[054D0100]                mov   esi, free_mem_p_str
2731 00004294 E89F000000                  call  bintdstr
2732                                pmim:
2733 00004299 BE[8B4C0100]                mov   esi, msg_memory_info
2734                                      ;
2735 0000429E B407                        mov   ah, 07h ; Black background,
2736                                            ; light gray forecolor
2737                                print_kmsg: ; 29/01/2016
2738 000042A0 8825[478A0100]              mov   [ccolor], ah
2739                                pkmsg_loop:
2740 000042A6 AC                          lodsb
2741 000042A7 08C0                        or    al, al
2742 000042A9 7410                        jz    short pkmsg_ok
2743 000042AB 56                          push  esi
2744                                      ; 13/05/2016
2745 000042AC 0FB61D[478A0100]            movzx ebx, byte [ccolor]
2746                                            ; Video page 0 (bh=0)
2747 000042B3 E84DE0FFFF                  call  _write_tty
2748 000042B8 5E                          pop   esi
2749 000042B9 EBEB                        jmp   short pkmsg_loop
2750                                pkmsg_ok:
2751 000042BB C3                          retn
2752
2753                                ; 19/12/2020
2754                                ; temporary
2755                                ; Write default liner frame buffer address
2756                                ;
2757                                default_lfb_info:
2758 000042BC 66A1[EB0E0000]              mov   ax, [def_LFB_addr] ; high word
2759 000042C2 E829000000                  call  wordtohex
2760 000042C7 A3[5D4D0100]                mov   dword [lfb_addr_str], eax
2761 000042CC BE[464D0100]                mov   esi, msg_lfb_addr
2762 000042D1 B40F                        mov   ah, 0Fh ; Black background,
2763                                            ; white forecolor
2764 000042D3 EBCB                        jmp   short print_kmsg
2765
2766                                ; Convert binary number to hexadecimal string
2767                                ; 10/05/2015
2768                                ; dsectpm.s (28/02/2015)
2769                                ; Retro UNIX 386 v1 - Kernel v0.2.0.6
2770                                ; 01/12/2014
2771                                ; 25/11/2014
2772                                ;
2773                                bytetohex:
2774                                      ; INPUT ->
```

```
2775                                    ;       AL = byte (binary number)
2776                                    ; OUTPUT ->
2777                                    ;       AX = hexadecimal string
2778                                    ;
2779 000042D5 53                       push    ebx
2780 000042D6 31DB                     xor     ebx, ebx
2781 000042D8 88C3                     mov     bl, al
2782 000042DA C0EB04                   shr     bl, 4
2783 000042DD 8A9B[27430000]           mov     bl, [ebx+hexchrs]
2784 000042E3 86D8                     xchg    bl, al
2785 000042E5 80E30F                   and     bl, 0Fh
2786 000042E8 8AA3[27430000]           mov     ah, [ebx+hexchrs]
2787 000042EE 5B                       pop     ebx
2788 000042EF C3                       retn
2789
2790                                wordtohex:
2791                                    ; INPUT ->
2792                                    ;       AX = word (binary number)
2793                                    ; OUTPUT ->
2794                                    ;       EAX = hexadecimal string
2795                                    ;
2796 000042F0 53                       push    ebx
2797 000042F1 31DB                     xor     ebx, ebx
2798 000042F3 86E0                     xchg    ah, al
2799 000042F5 6650                     push    ax
2800 000042F7 88E3                     mov     bl, ah
2801 000042F9 C0EB04                   shr     bl, 4
2802 000042FC 8A83[27430000]           mov     al, [ebx+hexchrs]
2803 00004302 88E3                     mov     bl, ah
2804 00004304 80E30F                   and     bl, 0Fh
2805 00004307 8AA3[27430000]           mov     ah, [ebx+hexchrs]
2806 0000430D C1E010                   shl     eax, 16
2807 00004310 6658                     pop     ax
2808 00004312 5B                       pop     ebx
2809 00004313 EBC0                     jmp     short bytetohex
2810                                    ;mov   bl, al
2811                                    ;shr   bl, 4
2812                                    ;mov   bl, [ebx+hexchrs]
2813                                    ;xchg  bl, al
2814                                    ;and   bl, 0Fh
2815                                    ;mov   ah, [ebx+hexchrs]
2816                                    ;pop   ebx
2817                                    ;retn
2818
2819                                dwordtohex:
2820                                    ; INPUT ->
2821                                    ;       EAX = dword (binary number)
2822                                    ; OUTPUT ->
2823                                    ;       EDX:EAX = hexadecimal string
2824                                    ;
2825 00004315 50                       push    eax
2826 00004316 C1E810                   shr     eax, 16
2827 00004319 E8D2FFFFFF               call    wordtohex
2828 0000431E 89C2                     mov     edx, eax
2829 00004320 58                       pop     eax
2830 00004321 E8CAFFFFFF               call    wordtohex
2831 00004326 C3                       retn
2832
2833                                ; 10/05/2015
2834                                hex_digits:
2835                                hexchrs:
2836 00004327 303132333435363738-       db '0123456789ABCDEF'
2836 00004330 39414243444546
2837                                ; 19/01/2021 - VESA EDID ready flag (4Fh)
2838 00004337 00                    edid: db 0
2839
2840                                ; Convert binary number to decimal/numeric string
2841                                ; 06/11/2014
2842                                ; Temporary Code
2843                                ;
2844
2845                                bintdstr:
2846                                    ; EAX = binary number
2847                                    ; ESI = decimal/numeric string address
2848                                    ; EBX = divisor (10)
2849                                    ; ECX = string length (<=10)
2850 00004338 01CE                     add     esi, ecx
2851                                btdstr0:
2852 0000433A 4E                       dec     esi
2853 0000433B 31D2                     xor     edx, edx
2854 0000433D F7F3                     div     ebx
2855 0000433F 80C230                   add     dl, 30h
2856 00004342 8816                     mov     [esi], dl
2857 00004344 FEC9                     dec     cl
2858 00004346 740C                     jz      short btdstr2 ; 08/09/2016
2859 00004348 09C0                     or      eax, eax
2860 0000434A 75EE                     jnz     short btdstr0
2861                                btdstr1:
2862 0000434C 4E                       dec     esi
2863 0000434D C60620                     mov     byte [esi], 20h ; blank space
2864 00004350 FEC9                     dec     cl
2865 00004352 75F8                     jnz     short btdstr1
2866                                btdstr2:
2867 00004354 C3                       retn
2868
2869                                ; Calculate free memory pages on M.A.T.
2870                                ; 06/11/2014
2871                                ; Temporary Code
2872                                ;
2873
2874                                calc_free_mem:
2875 00004355 31D2                     xor     edx, edx
2876                                    ;xor   ecx, ecx
2877 00004357 668B0D[308A0100]         mov     cx, [mat_size] ; in pages
2878 0000435E C1E10A                   shl     ecx, 10       ; 1024 dwords per page
```

```
2879 00004361 BE00001000                mov    esi, MEM_ALLOC_TBL
2880                                cfm0:
2881 00004366 AD                        lodsd
2882 00004367 51                        push   ecx
2883 00004368 B920000000                mov    ecx, 32
2884                                cfm1:
2885 0000436D D1E8                      shr    eax, 1
2886 0000436F 7301                      jnc    short cfm2
2887 00004371 42                        inc    edx
2888                                cfm2:
2889 00004372 E2F9                      loop   cfm1
2890 00004374 59                        pop    ecx
2891 00004375 E2EF                      loop   cfm0
2892 00004377 C3                        retn
2893
2894                                %include 'diskio.s'  ; 07/03/2015
   1                            <1> ; ****************************************************************************
   2                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.2 - diskio.s
   3                            <1> ; ----------------------------------------------------------------------------
   4                            <1> ; Last Update: 30/08/2020
   5                            <1> ; ----------------------------------------------------------------------------
   6                            <1> ; Beginning: 24/01/2016
   7                            <1> ; ----------------------------------------------------------------------------
   8                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                            <1> ; ----------------------------------------------------------------------------
  10                            <1> ; Turkish Rational DOS
  11                            <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                            <1> ;
  13                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  14                            <1> ; diskio.inc (22/08/2015)
  15                            <1> ;
  16                            <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
  17                            <1> ; ****************************************************************************
  18                            <1>
  19                            <1> ; Retro UNIX 386 v1 Kernel - DISKIO.INC
  20                            <1> ; Last Modification: 22/08/2015
  21                            <1> ;     (Initialized Disk Parameters Data is in 'DISKDATA.INC')
  22                            <1> ;     (Uninitialized Disk Parameters Data is in 'DISKBSS.INC')
  23                            <1>
  24                            <1> ; DISK I/O SYSTEM - Erdogan Tan (Retro UNIX 386 v1 project)
  25                            <1>
  26                            <1> ; ///////// DISK I/O SYSTEM ///////////////
  27                            <1>
  28                            <1> ; 06/02/2015
  29                            <1> diskette_io:
  30 00004378 F8                <1>       clc ; 20/07/2020
  31 00004379 9C                <1>       pushfd
  32 0000437A 0E                <1>       push   cs
  33 0000437B E809000000        <1>       call   DISKETTE_IO_1
  34 00004380 C3                <1>       retn
  35                            <1>
  36                            <1> ;;;;;; DISKETTE I/O ;;;;;;;;;;;;;;;;;;;; 06/02/2015 ;;;
  37                            <1> ;//////////////////////////////////////////////////////
  38                            <1>
  39                            <1> ; DISKETTE I/O - Erdogan Tan (Retro UNIX 386 v1 project)
  40                            <1> ; 20/02/2015
  41                            <1> ; 06/02/2015 (unix386.s)
  42                            <1> ; 16/12/2014 - 02/01/2015 (dsectrm2.s)
  43                            <1> ;
  44                            <1> ; Code (DELAY) modifications - AWARD BIOS 1999 (ADISK.EQU, COMMON.MAC)
  45                            <1> ;
  46                            <1> ; ADISK.EQU
  47                            <1>
  48                            <1> ;----- Wait control constants
  49                            <1>
  50                            <1> ;amount of time to wait while RESET is active.
  51                            <1>
  52                            <1> WAITCPU_RESET_ON   EQU    21          ;Reset on must last at least 14us
  53                            <1>                                       ;at 250 KBS xfer rate.
  54                            <1>                                       ;see INTEL MCS, 1985, pg. 5-456
  55                            <1>
  56                            <1> WAITCPU_FOR_STATUS EQU    100         ;allow 30 microseconds for
  57                            <1>                                       ;status register to become valid
  58                            <1>                                       ;before re-reading.
  59                            <1>
  60                            <1> ;After sending a byte to NEC, status register may remain
  61                            <1> ;incorrectly set for 24 us.
  62                            <1>
  63                            <1> WAITCPU_RQM_LOW          EQU    24          ;number of loops to check for
  64                            <1>                                       ;RQM low.
  65                            <1>
  66                            <1> ; COMMON.MAC
  67                            <1> ;
  68                            <1> ;     Timing macros
  69                            <1> ;
  70                            <1>
  71                            <1> %macro             SIODELAY 0                ; SHORT IODELAY
  72                            <1>             jmp short $+2
  73                            <1> %endmacro
  74                            <1>
  75                            <1> %macro             IODELAY  0                ; NORMAL IODELAY
  76                            <1>             jmp short $+2
  77                            <1>             jmp short $+2
  78                            <1> %endmacro
  79                            <1>
  80                            <1> %macro             NEWIODELAY 0
  81                            <1>             out    0ebh,al
  82                            <1> %endmacro
  83                            <1>
  84                            <1> ; (According to) AWARD BIOS 1999 - ATORGS.ASM (dw -> equ, db -> equ)
  85                            <1> ;;; WAIT_FOR_MEM
  86                            <1> ;WAIT_FDU_INT_LO   equ    017798       ; 2.5 secs in 30 micro units.
  87                            <1> ;WAIT_FDU_INT_HI   equ    1
  88                            <1> WAIT_FDU_INT_LH           equ    83334        ; 27/02/2015 (2.5 seconds waiting)
  89                            <1> ;;; WAIT_FOR_PORT
```

```
                        <1> ;WAIT_FDU_SEND_LO  equ    16667        ; .5 secons in 30 us units.
                        <1> ;WAIT_FDU_SEND_HI  equ    0
                        <1> WAIT_FDU_SEND_LH   equ    16667        ; 27/02/2015
                        <1> ;Time to wait while waiting for each byte of NEC results = .5
                        <1> ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
                        <1> ;WAIT_FDU_RESULTS_LO      equ    16667        ; .5 seconds in 30 micro units.
                        <1> ;WAIT_FDU_RESULTS_HI      equ    0
                        <1> WAIT_FDU_RESULTS_LH       equ    16667  ; 27/02/2015
                        <1> ;;; WAIT_REFRESH
                        <1> ;amount of time to wait for head settle, per unit in parameter
                        <1> ;table = 1 ms.
                        <1> WAIT_FDU_HEAD_SETTLE      equ    33           ; 1 ms in 30 micro units.
                        <1>
                        <1>
                        <1> ; ///////////////// DISKETTE I/O ////////////////
                        <1>
                        <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - POSTEQU.INC)
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     EQUATES USED BY POST AND BIOS    :
                        <1> ;----------------------------------------
                        <1>
                        <1> ;--------- 8042 KEYBOARD INTERFACE AND DIAGNOSTIC CONTROL REGISTERS ------------
                        <1> ;PORT_A            EQU    060H         ; 8042 KEYBOARD SCAN CODE/CONTROL PORT
                        <1> ;PORT_B            EQU    061H         ; PORT B READ/WRITE DIAGNOSTIC REGISTER
                        <1> ;REFRESH_BIT EQU    00010000B    ; REFRESH TEST BIT
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     CMOS EQUATES FOR THIS SYSTEM     :
                        <1> ;----------------------------------------------------------------------------
                        <1> ;CMOS_PORT   EQU    070H         ; I/O ADDRESS OF CMOS ADDRESS PORT
                        <1> ;CMOS_DATA   EQU    071H         ; I/O ADDRESS OF CMOS DATA PORT
                        <1> ;NMI         EQU    10000000B    ; DISABLE NMI INTERRUPTS MASK -
                        <1>                                  ;  HIGH BIT OF CMOS LOCATION ADDRESS
                        <1>
                        <1> ;---------- CMOS TABLE LOCATION ADDRESS'S ## ---------------------------------
                        <1> CMOS_DISKETTE      EQU    010H         ; DISKETTE DRIVE TYPE BYTE      ;
                        <1> ;           EQU    011H         ; - RESERVED                    ;C
                        <1> CMOS_DISK    EQU    012H         ; FIXED DISK TYPE BYTE                 ;H
                        <1> ;           EQU    013H         ; - RESERVED                    ;E
                        <1> CMOS_EQUIP   EQU    014H         ; EQUIPMENT WORD LOW BYTE       ;C
                        <1>
                        <1> ;---------- DISKETTE EQUATES -----------------------------------------------
                        <1> INT_FLAG     EQU    10000000B    ; INTERRUPT OCCURRENCE FLAG
                        <1> DSK_CHG      EQU    10000000B    ; DISKETTE CHANGE FLAG MASK BIT
                        <1> DETERMINED   EQU    00010000B    ; SET STATE DETERMINED IN STATE BITS
                        <1> HOME         EQU    00010000B    ; TRACK 0 MASK
                        <1> SENSE_DRV_ST EQU    00000100B    ; SENSE DRIVE STATUS COMMAND
                        <1> TRK_SLAP     EQU    030H         ; CRASH STOP (48 TPI DRIVES)
                        <1> QUIET_SEEK   EQU    00AH         ; SEEK TO TRACK 10
                        <1> ;MAX_DRV     EQU    2            ; MAX NUMBER OF DRIVES
                        <1> HD12_SETTLE  EQU    15           ; 1.2 M HEAD SETTLE TIME
                        <1> HD320_SETTLE EQU    20           ; 320 K HEAD SETTLE TIME
                        <1> MOTOR_WAIT   EQU    37           ; 2 SECONDS OF COUNTS FOR MOTOR TURN OFF
                        <1>
                        <1> ;---------- DISKETTE ERRORS -------------------------------------------------
                        <1> ;TIME_OUT    EQU    080H         ; ATTACHMENT FAILED TO RESPOND
                        <1> ;BAD_SEEK    EQU    040H         ; SEEK OPERATION FAILED
                        <1> BAD_NEC      EQU    020H         ; DISKETTE CONTROLLER HAS FAILED
                        <1> BAD_CRC      EQU    010H         ; BAD CRC ON DISKETTE READ
                        <1> MED_NOT_FND  EQU    00CH         ; MEDIA TYPE NOT FOUND
                        <1> DMA_BOUNDARY EQU    009H         ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
                        <1> BAD_DMA      EQU    008H         ; DMA OVERRUN ON OPERATION
                        <1> MEDIA_CHANGE EQU    006H         ; MEDIA REMOVED ON DUAL ATTACH CARD
                        <1> RECORD_NOT_FND     EQU    004H         ; REQUESTED SECTOR NOT FOUND
                        <1> WRITE_PROTECT      EQU    003H         ; WRITE ATTEMPTED ON WRITE PROTECT DISK
                        <1> BAD_ADDR_MARK      EQU    002H         ; ADDRESS MARK NOT FOUND
                        <1> BAD_CMD      EQU    001H         ; BAD COMMAND PASSED TO DISKETTE I/O
                        <1>
                        <1> ;---------- DISK CHANGE LINE EQUATES ----------------------------------------
                        <1> NOCHGLN      EQU    001H         ; NO DISK CHANGE LINE AVAILABLE
                        <1> CHGLN        EQU    002H         ; DISK CHANGE LINE AVAILABLE
                        <1>
                        <1> ;---------- MEDIA/DRIVE STATE INDICATORS ------------------------------------
                        <1> TRK_CAPA     EQU    00000001B    ; 80 TRACK CAPABILITY
                        <1> FMT_CAPA     EQU    00000010B    ; MULTIPLE FORMAT CAPABILITY (1.2M)
                        <1> DRV_DET      EQU    00000100B    ; DRIVE DETERMINED
                        <1> MED_DET      EQU    00010000B    ; MEDIA DETERMINED BIT
                        <1> DBL_STEP     EQU    00100000B    ; DOUBLE STEP BIT
                        <1> RATE_MSK     EQU    11000000B    ; MASK FOR CLEARING ALL BUT RATE
                        <1> RATE_500     EQU    00000000B    ; 500 KBS DATA RATE
                        <1> RATE_300     EQU    01000000B    ; 300 KBS DATA RATE
                        <1> RATE_250     EQU    10000000B    ; 250 KBS DATA RATE
                        <1> STRT_MSK     EQU    00001100B    ; OPERATION START RATE MASK
                        <1> SEND_MSK     EQU    11000000B    ; MASK FOR SEND RATE BITS
                        <1>
                        <1> ;---------- MEDIA/DRIVE STATE INDICATORS COMPATIBILITY ------------------------
                        <1> M3D3U        EQU    00000000B    ; 360 MEDIA/DRIVE NOT ESTABLISHED
                        <1> M3D1U        EQU    00000001B    ; 360 MEDIA,1.2DRIVE NOT ESTABLISHED
                        <1> M1D1U        EQU    00000010B    ; 1.2 MEDIA/DRIVE NOT ESTABLISHED
                        <1> MED_UNK      EQU    00000111B    ; NONE OF THE ABOVE
                        <1>
                        <1> ;---------- INTERRUPT EQUATES -----------------------------------------------
                        <1> ;EOI         EQU    020H         ; END OF INTERRUPT COMMAND TO 8259
                        <1> ;INTA00            EQU    020H         ; 8259 PORT
                        <1> INTA01             EQU    021H         ; 8259 PORT
                        <1> INTB00             EQU    0A0H         ; 2ND 8259
                        <1> INTB01             EQU    0A1H         ;
                        <1>
                        <1> ;----------------------------------------------------------------------------
                        <1> DMA08        EQU    008H         ; DMA STATUS REGISTER PORT ADDRESS
                        <1> DMA          EQU    000H         ; DMA CH.0 ADDRESS REGISTER PORT ADDRESS
                        <1> DMA18        EQU    0D0H         ; 2ND DMA STATUS PORT ADDRESS
                        <1> DMA1         EQU    0C0H         ; 2ND DMA CH.0 ADDRESS REGISTER ADDRESS
                        <1> ;----------------------------------------------------------------------------
```

```
195                             <1> ;TIMER             EQU    040H         ; 8254 TIMER - BASE ADDRESS
196                             <1>
197                             <1> ;--------------------------------------------------------------------------------
198                             <1> DMA_PAGE     EQU    081H         ; START OF DMA PAGE REGISTERS
199                             <1>
200                             <1> ; 06/02/2015 (unix386.s, protected mode modifications)
201                             <1> ; (unix386.s <-- dsectrm2.s)
202                             <1> ; 11/12/2014 (copy from IBM PC-XT Model 286 BIOS - DSEG.INC)
203                             <1>
204                             <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
205                             <1> ; 10/12/2014
206                             <1> ;
207                             <1> ;int40h:
208                             <1> ;     pushf
209                             <1> ;     push   cs
210                             <1> ;     ;cli
211                             <1> ;     call   DISKETTE_IO_1
212                             <1> ;     retn
213                             <1>
214                             <1> ; DSKETTE ----- 04/21/86 DISKETTE BIOS
215                             <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
216                             <1> ;
217                             <1>
218                             <1> ;-- INT13H ----------------------------------------------------------------------
219                             <1> ; DISKETTE I/O
220                             <1> ;     THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 INCH 360 KB,
221                             <1> ;     1.2 MB, 720 KB AND 1.44 MB DISKETTE DRIVES.
222                             <1> ; INPUT
223                             <1> ;     (AH) =  00H RESET DISKETTE SYSTEM
224                             <1> ;           HARD RESET TO NEC, PREPARE COMMAND, RECALIBRATE REQUIRED
225                             <1> ;           ON ALL DRIVES
226                             <1> ;--------------------------------------------------------------------------------
227                             <1> ;     (AH)= 01H  READ THE STATUS OF THE SYSTEM INTO (AH)
228                             <1> ;           @DISKETTE_STATUS FROM LAST OPERATION IS USED
229                             <1> ;--------------------------------------------------------------------------------
230                             <1> ;     REGISTERS FOR READ/WRITE/VERIFY/FORMAT
231                             <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
232                             <1> ;     (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
233                             <1> ;     (CH) - TRACK NUMBER (NOT VALUE CHECKED)
234                             <1> ;           MEDIA  DRIVE  TRACK NUMBER
235                             <1> ;           320/360      320/360          0-39
236                             <1> ;           320/360      1.2M       0-39
237                             <1> ;           1.2M   1.2M       0-79
238                             <1> ;           720K   720K       0-79
239                             <1> ;           1.44M  1.44M      0-79
240                             <1> ;     (CL) -       SECTOR NUMBER (NOT VALUE CHECKED, NOT USED FOR FORMAT)
241                             <1> ;           MEDIA  DRIVE  SECTOR NUMBER
242                             <1> ;           320/360      320/360           1-8/9
243                             <1> ;           320/360      1.2M        1-8/9
244                             <1> ;           1.2M   1.2M       1-15
245                             <1> ;           720K   720K       1-9
246                             <1> ;           1.44M  1.44M      1-18
247                             <1> ;     (AL)   NUMBER OF SECTORS (NOT VALUE CHECKED)
248                             <1> ;           MEDIA  DRIVE  MAX NUMBER OF SECTORS
249                             <1> ;           320/360      320/360              8/9
250                             <1> ;           320/360      1.2M           8/9
251                             <1> ;           1.2M   1.2M         15
252                             <1> ;           720K   720K         9
253                             <1> ;           1.44M  1.44M        18
254                             <1> ;
255                             <1> ;     (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY)
256                             <1> ;
257                             <1> ;--------------------------------------------------------------------------------
258                             <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY
259                             <1> ;--------------------------------------------------------------------------------
260                             <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY
261                             <1> ;--------------------------------------------------------------------------------
262                             <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS
263                             <1> ;--------------------------------------------------------------------------------
264                             <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK
265                             <1> ;           (ES,BX) MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
266                             <1> ;           FOR THE      TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, (C,H,R,N),
267                             <1> ;           WHERE C = TRACK NUMBER, H=HEAD NUMBER, R = SECTOR NUMBER,
268                             <1> ;           N= NUMBER OF BYTES PER SECTOR (00=128,01=256,02=512,03=1024),
269                             <1> ;           THERE MUST BE ONE ENTRY FOR EVERY SECTOR ON THE TRACK.
270                             <1> ;           THIS INFORMATION IS USED TO FIND THE REQUESTED SECTOR DURING
271                             <1> ;           READ/WRITE ACCESS.
272                             <1> ;           PRIOR TO FORMATTING A DISKETTE, IF THERE EXISTS MORE THAN
273                             <1> ;           ONE SUPPORTED MEDIA FORMAT TYPE WITHIN THE DRIVE IN QUESTION,
274                             <1> ;           THEN "SET DASD TYPE" (INT 13H, AH = 17H) OR 'SET MEDIA TYPE'
275                             <1> ;           (INT 13H, AH =  18H) MUST BE CALLED TO SET THE DISKETTE TYPE
276                             <1> ;           THAT IS TO BE FORMATTED. IF "SET DASD TYPE" OR "SET MEDIA TYPE"
277                             <1> ;           IS NOT CALLED, THE FORMAT ROUTINE WILL ASSUME THE
278                             <1> ;           MEDIA FORMAT TO BE THE MAXIMUM CAPACITY OF THE DRIVE.
279                             <1> ;
280                             <1> ;           THESE PARAMETERS OF DISK BASE MUST BE CHANGED IN ORDER TO
281                             <1> ;           FORMAT THE FOLLOWING MEDIAS:
282                             <1> ;           ---------------------------------------------
283                             <1> ;           : MEDIA  :     DRIVE      : PARM 1 : PARM 2 :
284                             <1> ;           ---------------------------------------------
285                             <1> ;           : 320K  : 320K/360K/1.2M :  50H   :   8    :
286                             <1> ;           : 360K  : 320K/360K/1.2M :  50H   :   9    :
287                             <1> ;           : 1.2M  : 1.2M           :  54H   :  15    :
288                             <1> ;           : 720K  : 720K/1.44M     :  50H   :   9    :
289                             <1> ;           : 1.44M       : 1.44M          :  6CH   :  18    :
290                             <1> ;           ---------------------------------------------
291                             <1> ;           NOTES: - PARM 1 = GAP LENGTH FOR FORMAT
292                             <1> ;                  - PARM 2 = EOT (LAST SECTOR ON TRACK)
293                             <1> ;                  - DISK BASE IS POINTED BY DISK POINTER LOCATED
294                             <1> ;                   AT ABSOLUTE ADDRESS 0:78.
295                             <1> ;                  - WHEN FORMAT OPERATIONS ARE COMPLETE, THE PARAMETERS
296                             <1> ;                   SHOULD BE RESTORED TO THEIR RESPECTIVE INITIAL VALUES.
297                             <1> ;--------------------------------------------------------------------------------
298                             <1> ;     (AH) = 08H READ DRIVE PARAMETERS
299                             <1> ;     REGISTERS
```

```
300                                     <1> ;       INPUT
301                                     <1> ;         (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
302                                     <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
303                                     <1> ;            ** EBX = Buffer address for floppy disk parameters table **
304                                     <1> ;       OUTPUT
305                                     <1> ;         (ES:DI) POINTS TO DRIVE PARAMETER TABLE
306                                     <1> ;         *** TRDOS 386 note: floppy disk parameter table (16 bytes)
307                                     <1> ;         will be returned to user in EBX, buffer address *** 27/05/2016 ***
308                                     <1> ;
309                                     <1> ;         (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM NUMBER OF TRACKS
310                                     <1> ;         (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
311                                     <1> ;                BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
312                                     <1> ;         (DH) - MAXIMUM HEAD NUMBER
313                                     <1> ;         (DL) - NUMBER OF DISKETTE DRIVES INSTALLED
314                                     <1> ;         (BH) - 0
315                                     <1> ;         (BL) - BITS 7 THRU 4 - 0
316                                     <1> ;                BITS 3 THRU 0 - VALID DRIVE TYPE VALUE IN CMOS
317                                     <1> ;         (AX) - 0
318                                     <1> ;      UNDER THE FOLLOWING CIRCUMSTANCES:
319                                     <1> ;         (1) THE DRIVE NUMBER IS INVALID,
320                                     <1> ;         (2) THE DRIVE TYPE IS UNKNOWN AND CMOS IS NOT PRESENT,
321                                     <1> ;         (3) THE DRIVE TYPE IS UNKNOWN AND CMOS IS BAD,
322                                     <1> ;         (4) OR THE DRIVE TYPE IS UNKNOWN AND THE CMOS DRIVE TYPE IS INVALID
323                                     <1> ;         THEN ES,AX,BX,CX,DH,DI=0 ; DL=NUMBER OF DRIVES.
324                                     <1> ;         IF NO DRIVES ARE PRESENT THEN: ES,AX,BX,CX,DX,DI=0.
325                                     <1> ;         @DISKETTE_STATUS = 0 AND CY IS RESET.
326                                     <1> ;--------------------------------------------------------------------------------
327                                     <1> ;     (AH)= 15H  READ DASD TYPE
328                                     <1> ;     OUTPUT REGISTERS
329                                     <1> ;     (AH) - ON RETURN IF CARRY FLAG NOT SET, OTHERWISE ERROR
330                                     <1> ;            00 - DRIVE NOT PRESENT
331                                     <1> ;            01 - DISKETTE, NO CHANGE LINE AVAILABLE
332                                     <1> ;            02 - DISKETTE, CHANGE LINE AVAILABLE
333                                     <1> ;            03 - RESERVED (FIXED DISK)
334                                     <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
335                                     <1> ;--------------------------------------------------------------------------------
336                                     <1> ;     (AH)= 16H  DISK CHANGE LINE STATUS
337                                     <1> ;     OUTPUT REGISTERS
338                                     <1> ;     (AH) - 00 - DISK CHANGE LINE NOT ACTIVE
339                                     <1> ;            06 - DISK CHANGE LINE ACTIVE & CARRY BIT ON
340                                     <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED)
341                                     <1> ;--------------------------------------------------------------------------------
342                                     <1> ;     (AH)= 17H  SET DASD TYPE FOR FORMAT
343                                     <1> ;     INPUT REGISTERS
344                                     <1> ;     (AL) - 00 - NOT USED
345                                     <1> ;            01 - DISKETTE 320/360K IN 360K DRIVE
346                                     <1> ;            02 - DISKETTE 360K IN 1.2M DRIVE
347                                     <1> ;            03 - DISKETTE 1.2M IN 1.2M DRIVE
348                                     <1> ;            04 - DISKETTE 720K IN 720K DRIVE
349                                     <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHECKED:
350                                     <1> ;             (DO NOT USE WHEN DISKETTE ATTACH CARD USED)
351                                     <1> ;--------------------------------------------------------------------------------
352                                     <1> ;     (AH)= 18H  SET MEDIA TYPE FOR FORMAT
353                                     <1> ;     INPUT REGISTERS
354                                     <1> ;     (CH) - LOW ORDER 8 OF 10 BITS MAXIMUM TRACKS
355                                     <1> ;     (CL) - BITS 7 & 6 - HIGH ORDER TWO BITS OF MAXIMUM TRACKS
356                                     <1> ;            BITS 5 THRU 0 - MAXIMUM SECTORS PER TRACK
357                                     <1> ;     (DL) - DRIVE NUMBER (0-1 ALLOWED, VALUE CHACKED)
358                                     <1> ;     OUTPUT REGISTERS:
359                                     <1> ;     (ES:DI) - POINTER TO DRIVE PARAMETERS TABLE FOR THIS MEDIA TYPE,
360                                     <1> ;              UNCHANGED IF (AH) IS NON-ZERO
361                                     <1> ;     (AH) - 00H, CY = 0, TRACK AND SECTORS/TRACK COMBINATION IS SUPPORTED
362                                     <1> ;          - 01H, CY = 1, FUNCTION IS NOT AVAILABLE
363                                     <1> ;          - 0CH, CY = 1, TRACK AND SECTORS/TRACK COMBINATION IS NOT SUPPORTED
364                                     <1> ;          - 80H, CY = 1, TIME OUT (DISKETTE NOT PRESENT)
365                                     <1> ;--------------------------------------------------------------------------------
366                                     <1> ;     DISK CHANGE STATUS IS ONLY CHECKED WHEN A MEDIA SPECIFIED IS OTHER
367                                     <1> ;     THAN 360 KB DRIVE. IF THE DISK CHANGE LINE IS FOUND TO BE
368                                     <1> ;     ACTIVE THE FOLLOWING ACTIONS TAKE PLACE:
369                                     <1> ;           ATTEMPT TO RESET DISK CHANGE LINE TO INACTIVE STATE.
370                                     <1> ;           IF ATTEMPT SUCCEEDS SET DASD TYPE FOR FORMAT AND RETURN DISK
371                                     <1> ;           CHANGE ERROR CODE
372                                     <1> ;           IF ATTEMPT FAILS RETURN TIMEOUT ERROR CODE AND SET DASD TYPE
373                                     <1> ;           TO A PREDETERMINED STATE INDICATING MEDIA TYPE UNKNOWN.
374                                     <1> ;     IF THE DISK CHANGE LINE IN INACTIVE PERFORM SET DASD TYPE FOR FORMAT.
375                                     <1> ;
376                                     <1> ; DATA VARIABLE -- @DISK_POINTER
377                                     <1> ;     DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS
378                                     <1> ;--------------------------------------------------------------------------------
379                                     <1> ; OUTPUT FOR ALL FUNCTIONS
380                                     <1> ;     AH = STATUS OF OPERATION
381                                     <1> ;           STATUS BITS ARE DEFINED IN THE EQUATES FOR @DISKETTE_STATUS
382                                     <1> ;           VARIABLE IN THE DATA SEGMENT OF THIS MODULE
383                                     <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN, EXCEPT FOR READ DASD
384                                     <1> ;           TYPE AH=(15)).
385                                     <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)
386                                     <1> ;     FOR READ/WRITE/VERIFY
387                                     <1> ;           DS,BX,DX,CX PRESERVED
388                                     <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE APPROPRIATE
389                                     <1> ;           ACTION IS TO RESET THE DISKETTE, THEN RETRY THE OPERATION.
390                                     <1> ;           ON READ ACCESSES, NO MOTOR START DELAY IS TAKEN, SO THAT
391                                     <1> ;           THREE RETRIES ARE REQUIRED ON READS TO ENSURE THAT THE
392                                     <1> ;           PROBLEM IS NOT DUE TO MOTOR START-UP.
393                                     <1> ;--------------------------------------------------------------------------------
394                                     <1> ;
395                                     <1> ; DISKETTE STATE MACHINE - ABSOLUTE ADDRESS 40:90 (DRIVE A) & 91 (DRIVE B)
396                                     <1> ;
397                                     <1> ;   ---------------------------------------------------------------
398                                     <1> ;   |       |       |       |       |       |       |       |       |
399                                     <1> ;   |   7   |   6   |   5   |   4   |   3   |   2   |   1   |   0   |
400                                     <1> ;   |       |       |       |       |       |       |       |       |
401                                     <1> ;   ---------------------------------------------------------------
402                                     <1> ;     |       |       |       |       |       |       |       |
403                                     <1> ;     |       |       |       |       |     -----------------
404                                     <1> ;     |       |       |       |       |             |
```

```
                                <1> ;     |     |     |     |    RESERVED      |
                                <1> ;     |     |     |     |              PRESENT STATE
                                <1> ;     |     |     |     |      000: 360K IN 360K DRIVE UNESTABLISHED
                                <1> ;     |     |     |     |      001: 360K IN 1.2M DRIVE UNESTABLISHED
                                <1> ;     |     |     |     |      010: 1.2M IN 1.2M DRIVE UNESTABLISHED
                                <1> ;     |     |     |     |      011: 360K IN 360K DRIVE ESTABLISHED
                                <1> ;     |     |     |     |      100: 360K IN 1.2M DRIVE ESTABLISHED
                                <1> ;     |     |     |     |      101: 1.2M IN 1.2M DRIVE ESTABLISHED
                                <1> ;     |     |     |     |      110: RESERVED
                                <1> ;     |     |     |     |      111: NONE OF THE ABOVE
                                <1> ;     |     |     |     |
                                <1> ;     |     |     |     ------>      MEDIA/DRIVE ESTABLISHED
                                <1> ;     |     |     |
                                <1> ;     |     |     --------------->     DOUBLE STEPPING REQUIRED (360K IN 1.2M
                                <1> ;     |     |                   DRIVE)
                                <1> ;     |     |
                                <1> ;     ------------------------------>  DATA TRANSFER RATE FOR THIS DRIVE:
                                <1> ;
                                <1> ;                                       00: 500 KBS
                                <1> ;                                       01: 300 KBS
                                <1> ;                                       10: 250 KBS
                                <1> ;                                       11: RESERVED
                                <1> ;
                                <1> ;
                                <1> ;-----------------------------------------------------------------------------
                                <1> ; STATE OPERATION STARTED - ABSOLUTE ADDRESS 40:92 (DRIVE A) & 93 (DRIVE B)
                                <1> ;-----------------------------------------------------------------------------
                                <1> ; PRESENT CYLINDER NUMBER - ABSOLUTE ADDRESS 40:94 (DRIVE A) & 95 (DRIVE B)
                                <1> ;-----------------------------------------------------------------------------
                                <1>
                                <1> struc MD
00000000 <res 00000001>         <1>       .SPEC1       resb   1      ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00000001 <res 00000001>         <1>       .SPEC2       resb   1      ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00000002 <res 00000001>         <1>       .OFF_TIM     resb   1      ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00000003 <res 00000001>         <1>       .BYT_SEC     resb   1      ; 512 BYTES/SECTOR
00000004 <res 00000001>         <1>       .SEC_TRK     resb   1      ; EOT (LAST SECTOR ON TRACK)
00000005 <res 00000001>         <1>       .GAP         resb   1      ; GAP LENGTH
00000006 <res 00000001>         <1>       .DTL         resb   1      ; DTL
00000007 <res 00000001>         <1>       .GAP3        resb   1      ; GAP LENGTH FOR FORMAT
00000008 <res 00000001>         <1>       .FIL_BYT     resb   1      ; FILL BYTE FOR FORMAT
00000009 <res 00000001>         <1>       .HD_TIM              resb   1      ; HEAD SETTLE TIME (MILLISECONDS)
0000000A <res 00000001>         <1>       .STR_TIM     resb   1      ; MOTOR START TIME (1/8 SECONDS)
0000000B <res 00000001>         <1>       .MAX_TRK     resb   1      ; MAX. TRACK NUMBER
0000000C <res 00000001>         <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:
00004381 9C                     <1>       pushfd
00004382 0E                     <1>       push   cs
00004383 E801000000             <1>       call   DISKETTE_IO_1
00004388 C3                     <1>       retn
                                <1>
                                <1> DISKETTE_IO_1:
                                <1>
00004389 FB                     <1>       STI                         ; INTERRUPTS BACK ON
0000438A 55                     <1>       PUSH   eBP                  ; USER REGISTER
0000438B 57                     <1>       PUSH   eDI                  ; USER REGISTER
0000438C 52                     <1>       PUSH   eDX                  ; HEAD #, DRIVE # OR USER REGISTER
0000438D 53                     <1>       PUSH   eBX                  ; BUFFER OFFSET PARAMETER OR REGISTER
0000438E 51                     <1>       PUSH   eCX                  ; TRACK #-SECTOR # OR USER REGISTER
0000438F 89E5                   <1>       MOV    eBP,eSP                    ; BP     => PARAMETER LIST DEP. ON AH
                                <1>                                   ; [BP]   = SECTOR #
                                <1>                                   ; [BP+1] = TRACK #
                                <1>                                   ; [BP+2] = BUFFER OFFSET
                                <1>                                   ; FOR RETURN OF DRIVE PARAMETERS:
                                <1>                                   ; CL/[BP] = BITS 7&6 HI BITS OF MAX CYL
                                <1>                                   ;         BITS 0-5 MAX SECTORS/TRACK
                                <1>                                   ; CH/[BP+1] = LOW 8 BITS OF MAX CYL.
                                <1>                                   ; BL/[BP+2] = BITS 7-4 = 0
                                <1>                                   ;           BITS 3-0 = VALID CMOS TYPE
                                <1>                                   ; BH/[BP+3] = 0
                                <1>                                   ; DL/[BP+4] = # DRIVES INSTALLED
                                <1>                                   ; DH/[BP+5] = MAX HEAD #
                                <1>                                   ; DI/[BP+6] = OFFSET TO DISK BASE
00004391 06                     <1>       push   es ; 06/02/2015
00004392 1E                     <1>       PUSH   DS                   ; BUFFER SEGMENT PARM OR USER REGISTER
00004393 56                     <1>       PUSH   eSI                  ; USER REGISTERS
                                <1>       ;CALL  DDS                  ; SEGMENT OF BIOS DATA AREA TO DS
                                <1>       ;mov   cx, cs
                                <1>       ;mov   ds, cx
00004394 66B91000               <1>       mov    cx, KDATA
00004398 8ED9                   <1>         mov     ds, cx
0000439A 8EC1                   <1>         mov     es, cx
                                <1>
                                <1>       ;CMP   AH,(FNC_TAE-FNC_TAB)/2    ; CHECK FOR > LARGEST FUNCTION
0000439C 80FC19                 <1>       cmp    ah,(FNC_TAE-FNC_TAB)/4  ; 18/02/2015
0000439F 7202                   <1>       JB     short OK_FUNC        ; FUNCTION OK
000043A1 B414                   <1>       MOV    AH,14H               ; REPLACE WITH KNOWN INVALID FUNCTION
                                <1> OK_FUNC:
000043A3 80FC01                 <1>       CMP    AH,1                 ; RESET OR STATUS ?
000043A6 760C                   <1>       JBE    short OK_DRV         ; IF RESET OR STATUS DRIVE ALWAYS OK
000043A8 80FC08                 <1>       CMP    AH,8                 ; READ DRIVE PARMS ?
000043AB 7407                   <1>       JZ     short OK_DRV         ; IF SO DRIVE CHECKED LATER
000043AD 80FA01                 <1>       CMP    DL,1                 ; DRIVES 0 AND 1 OK
000043B0 7602                   <1>       JBE    short OK_DRV         ; IF 0 OR 1 THEN JUMP
000043B2 B414                   <1>       MOV    AH,14H               ; REPLACE WITH KNOWN INVALID FUNCTION
                                <1> OK_DRV:
000043B4 31C9                   <1>       xor    ecx, ecx
```

```
                                <1>       ;mov   esi, ecx ; 08/02/2015
000043B6 89CF                   <1>       mov    edi, ecx ; 08/02/2015
000043B8 88E1                   <1>       MOV    CL,AH               ; CL = FUNCTION
                                <1>       ;XOR   CH,CH               ; CX = FUNCTION
                                <1>       ;SHL   CL, 1               ; FUNCTION TIMES 2
000043BA C0E102                 <1>       SHL    CL, 2 ; 20/02/2015  ; FUNCTION TIMES 4 (for 32 bit offset)
000043BD BB[F5430000]           <1>       MOV    eBX,FNC_TAB         ; LOAD START OF FUNCTION TABLE
000043C2 01CB                   <1>       ADD    eBX,eCX                   ; ADD OFFSET INTO TABLE => ROUTINE
000043C4 88F4                   <1>       MOV    AH,DH               ; AX = HEAD #,# OF SECTORS OR DASD TYPE
000043C6 30F6                   <1>       XOR    DH,DH               ; DX = DRIVE #
000043C8 6689C6                 <1>       MOV    SI,AX               ; SI = HEAD #,# OF SECTORS OR DASD TYPE
000043CB 6689D7                 <1>       MOV     DI,DX                   ; DI = DRIVE #
                                <1>       ;
                                <1>       ; 11/12/2014
000043CE 8815[FD6D0000]         <1>         mov     [cfd], dl               ; current floppy drive (for 'GET_PARM')
                                <1>       ;
000043D4 8A25[A08A0100]         <1>       MOV    AH, [DSKETTE_STATUS]       ; LOAD STATUS TO AH FOR STATUS FUNCTION
000043DA C605[A08A0100]00       <1>       MOV    byte [DSKETTE_STATUS],0    ; INITIALIZE FOR ALL OTHERS
                                <1>
                                <1> ;     THROUGHOUT THE DISKETTE BIOS, THE FOLLOWING INFORMATION IS CONTAINED IN
                                <1> ;     THE FOLLOWING MEMORY LOCATIONS AND REGISTERS. NOT ALL DISKETTE BIOS
                                <1> ;     FUNCTIONS REQUIRE ALL OF THESE PARAMETERS.
                                <1> ;
                                <1> ;           DI     : DRIVE #
                                <1> ;           SI-HI  : HEAD #
                                <1> ;           SI-LOW : # OF SECTORS OR DASD TYPE FOR FORMAT
                                <1> ;           ES     : BUFFER SEGMENT
                                <1> ;           [BP]   : SECTOR #
                                <1> ;           [BP+1] : TRACK #
                                <1> ;           [BP+2] : BUFFER OFFSET
                                <1> ;
                                <1> ;     ACROSS CALLS TO SUBROUTINES THE CARRY FLAG (CY=1), WHERE INDICATED IN
                                <1> ;     SUBROUTINE PROLOGUES, REPRESENTS AN EXCEPTION RETURN (NORMALLY AN ERROR
                                <1> ;     CONDITION). IN MOST CASES, WHEN CY = 1, @DSKETTE_STATUS CONTAINS THE
                                <1> ;     SPECIFIC ERROR CODE.
                                <1> ;
                                <1>                                   ; (AH) = @DSKETTE_STATUS
000043E1 FF13                   <1>       CALL   dWORD [eBX]          ; CALL THE REQUESTED FUNCTION
000043E3 5E                     <1>       POP    eSI                  ; RESTORE ALL REGISTERS
000043E4 1F                     <1>       POP    DS
000043E5 07                     <1>       pop    es     ; 06/02/2015
000043E6 59                     <1>       POP    eCX
000043E7 5B                     <1>       POP    eBX
000043E8 5A                     <1>       POP    eDX
000043E9 5F                     <1>       POP    eDI
000043EA 89E5                   <1>       MOV    eBP, eSP
000043EC 50                     <1>       PUSH   eAX
000043ED 9C                     <1>       PUSHFd
000043EE 58                     <1>       POP    eAX
                                <1>       ;MOV   [BP+6], AX
000043EF 89450C                 <1>       mov    [ebp+12], eax  ; 18/02/2015, flags
000043F2 58                     <1>       POP    eAX
000043F3 5D                     <1>       POP    eBP
000043F4 CF                     <1>       IRETd
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; DW --> dd (06/02/2015)
000043F5 [59440000]             <1> FNC_TAB      dd     DSK_RESET           ; AH = 00H; RESET
000043F9 [D2440000]             <1>       dd     DSK_STATUS          ; AH = 01H; STATUS
000043FD [E3440000]             <1>       dd     DSK_READ            ; AH = 02H; READ
00004401 [F4440000]             <1>       dd     DSK_WRITE           ; AH = 03H; WRITE
00004405 [05450000]             <1>       dd     DSK_VERF            ; AH = 04H; VERIFY
00004409 [16450000]             <1>       dd     DSK_FORMAT          ; AH = 05H; FORMAT
0000440D [9B450000]             <1>       dd     FNC_ERR                   ; AH = 06H; INVALID
00004411 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 07H; INVALID
00004415 [A8450000]             <1>       dd     DSK_PARMS           ; AH = 08H; READ DRIVE PARAMETERS
00004419 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 09H; INVALID
0000441D [9B450000]             <1>       dd     FNC_ERR                   ; AH = 0AH; INVALID
00004421 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 0BH; INVALID
00004425 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 0CH; INVALID
00004429 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 0DH; INVALID
0000442D [9B450000]             <1>       dd     FNC_ERR                   ; AH = 0EH; INVALID
00004431 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 0FH; INVALID
00004435 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 10H; INVALID
00004439 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 11H; INVALID
0000443D [9B450000]             <1>       dd     FNC_ERR                   ; AH = 12H; INVALID
00004441 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 13H; INVALID
00004445 [9B450000]             <1>       dd     FNC_ERR                   ; AH = 14H; INVALID
00004449 [98460000]             <1>       dd     DSK_TYPE            ; AH = 15H; READ DASD TYPE
0000444D [C8460000]             <1>       dd     DSK_CHANGE          ; AH = 16H; CHANGE STATUS
00004451 [02470000]             <1>       dd     FORMAT_SET          ; AH = 17H; SET DASD TYPE
00004455 [85470000]             <1>       dd     SET_MEDIA           ; AH = 18H; SET MEDIA TYPE
                                <1> FNC_TAE EQU     $                    ; END
                                <1>
                                <1> ;--------------------------------------------------------------------------
                                <1> ; DISK_RESET (AH = 00H)
                                <1> ;           RESET THE DISKETTE SYSTEM.
                                <1> ;
                                <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                                <1> ;--------------------------------------------------------------------------
                                <1> DSK_RESET:
00004459 66BAF203               <1>       MOV    DX,03F2H            ; ADAPTER CONTROL PORT
0000445D FA                     <1>       CLI                        ; NO INTERRUPTS
0000445E A0[9E8A0100]           <1>       MOV    AL,[MOTOR_STATUS]   ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004463 243F                   <1>       AND    AL,00111111B        ; KEEP SELECTED AND MOTOR ON BITS
00004465 C0C004                 <1>       ROL    AL,4                ; MOTOR VALUE TO HIGH NIBBLE
                                <1>                                   ; DRIVE SELECT TO LOW NIBBLE
00004468 0C08                   <1>       OR     AL,00001000B        ; TURN ON INTERRUPT ENABLE
0000446A EE                     <1>       OUT    DX,AL               ; RESET THE ADAPTER
0000446B C605[9D8A0100]00       <1>       MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
                                <1>       ;JMP   $+2                 ; WAIT FOR I/O
                                <1>       ;JMP   $+2                 ; WAIT FOR I/O (TO INSURE MINIMUM
                                <1>                                   ;      PULSE WIDTH)
                                <1>       ; 19/12/2014
                                <1>       NEWIODELAY
```

```
614 00004472 E6EB               <2>  out 0ebh,al
615                             <1>
616                             <1>       ; 17/12/2014
617                             <1>       ; AWARD BIOS 1999 - RESETDRIVES (ADISK.ASM)
618 00004474 B915000000         <1>       mov    ecx, WAITCPU_RESET_ON     ; cx = 21 -- Min. 14 micro seconds !?
619                             <1> wdw1:
620                             <1>       NEWIODELAY   ; 27/02/2015
620 00004479 E6EB               <2>  out 0ebh,al
621 0000447B E2FC               <1>       loop   wdw1
622                             <1>       ;
623 0000447D 0C04               <1>       OR     AL,00000100B       ; TURN OFF RESET BIT
624 0000447F EE                 <1>       OUT    DX,AL              ; RESET THE ADAPTER
625                             <1>       ; 16/12/2014
626                             <1>       IODELAY
626 00004480 EB00               <2>  jmp short $+2
626 00004482 EB00               <2>  jmp short $+2
627                             <1>       ;
628                             <1>       ;STI                      ; ENABLE THE INTERRUPTS
629 00004484 E8590C0000         <1>       CALL   WAIT_INT           ; WAIT FOR THE INTERRUPT
630 00004489 723E               <1>       JC     short DR_ERR       ; IF ERROR, RETURN IT
631 0000448B 66B9C000           <1>       MOV    CX,11000000B       ; CL = EXPECTED @NEC_STATUS
632                             <1> NXT_DRV:
633 0000448F 6651               <1>       PUSH   CX                 ; SAVE FOR CALL
634 00004491 B8[C7440000]       <1>       MOV    eAX, DR_POP_ERR    ; LOAD NEC_OUTPUT ERROR ADDRESS
635 00004496 50                 <1>       PUSH   eAX                ; "
636 00004497 B408               <1>       MOV    AH,08H             ; SENSE INTERRUPT STATUS COMMAND
637 00004499 E8370B0000         <1>       CALL   NEC_OUTPUT
638 0000449E 58                 <1>       POP    eAX                ; THROW AWAY ERROR RETURN
639 0000449F E86E0C0000         <1>       CALL   RESULTS                 ; READ IN THE RESULTS
640 000044A4 6659               <1>       POP    CX                 ; RESTORE AFTER CALL
641 000044A6 7221               <1>       JC     short DR_ERR       ; ERROR RETURN
642 000044A8 3A0D[A18A0100]     <1>       CMP    CL, [NEC_STATUS]   ; TEST FOR DRIVE READY TRANSITION
643 000044AE 7519               <1>       JNZ    short DR_ERR       ; EVERYTHING OK
644 000044B0 FEC1               <1>       INC    CL                 ; NEXT EXPECTED @NEC_STATUS
645 000044B2 80F9C3             <1>       CMP    CL,11000011B       ; ALL POSSIBLE DRIVES CLEARED
646 000044B5 76D8               <1>       JBE    short NXT_DRV      ; FALL THRU IF 11000100B OR >
647                             <1>       ;
648 000044B7 E886030000         <1>       CALL   SEND_SPEC          ; SEND SPECIFY COMMAND TO NEC
649                             <1> RESBAC:
650 000044BC E83A090000         <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
651 000044C1 6689F3             <1>       MOV    BX,SI              ; GET SAVED AL TO BL
652 000044C4 88D8               <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
653 000044C6 C3                 <1>       RETn
654                             <1> DR_POP_ERR:
655 000044C7 6659               <1>       POP    CX                 ; CLEAR STACK
656                             <1> DR_ERR:
657 000044C9 800D[A08A0100]20   <1>       OR     byte [DSKETTE_STATUS],BAD_NEC ; SET ERROR CODE
658 000044D0 EBEA               <1>       JMP    SHORT RESBAC       ; RETURN FROM RESET
659                             <1>
660                             <1> ;--------------------------------------------------------------------------
661                             <1> ; DISK_STATUS      (AH = 01H)
662                             <1> ;     DISKETTE STATUS.
663                             <1> ;
664                             <1> ; ON ENTRY:  AH : STATUS OF PREVIOUS OPERATION
665                             <1> ;
666                             <1> ; ON EXIT:   AH, @DSKETTE_STATUS, CY REFLECT STATUS OF PREVIOUS OPERATION.
667                             <1> ;--------------------------------------------------------------------------
668                             <1> DSK_STATUS:
669 000044D2 8825[A08A0100]     <1>       MOV    [DSKETTE_STATUS],AH ; PUT BACK FOR SETUP END
670 000044D8 E81E090000         <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
671 000044DD 6689F3             <1>       MOV    BX,SI              ; GET SAVED AL TO BL
672 000044E0 88D8               <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
673 000044E2 C3                 <1>       RETn
674                             <1>
675                             <1> ;--------------------------------------------------------------------------
676                             <1> ; DISK_READ  (AH = 02H)
677                             <1> ;     DISKETTE READ.
678                             <1> ;
679                             <1> ; ON ENTRY:  DI     : DRIVE #
680                             <1> ;            SI-HI  : HEAD #
681                             <1> ;            SI-LOW : # OF SECTORS
682                             <1> ;            ES     : BUFFER SEGMENT
683                             <1> ;            [BP]   : SECTOR #
684                             <1> ;            [BP+1] : TRACK #
685                             <1> ;            [BP+2] : BUFFER OFFSET
686                             <1> ;
687                             <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
688                             <1> ;--------------------------------------------------------------------------
689                             <1>
690                             <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
691                             <1>
692                             <1> DSK_READ:
693 000044E3 8025[9E8A0100]7F   <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
694 000044EA 66B846E6           <1>       MOV    AX,0E646H          ; AX = NEC COMMAND, DMA COMMAND
695 000044EE E859040000         <1>       CALL   RD_WR_VF           ; COMMON READ/WRITE/VERIFY
696 000044F3 C3                 <1>       RETn
697                             <1>
698                             <1> ;--------------------------------------------------------------------------
699                             <1> ; DISK_WRITE (AH = 03H)
700                             <1> ;     DISKETTE WRITE.
701                             <1> ;
702                             <1> ; ON ENTRY:  DI     : DRIVE #
703                             <1> ;            SI-HI  : HEAD #
704                             <1> ;            SI-LOW : # OF SECTORS
705                             <1> ;            ES     : BUFFER SEGMENT
706                             <1> ;            [BP]   : SECTOR #
707                             <1> ;            [BP+1] : TRACK #
708                             <1> ;            [BP+2] : BUFFER OFFSET
709                             <1> ;
710                             <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
711                             <1> ;--------------------------------------------------------------------------
712                             <1>
713                             <1> ; 06/02/2015, ES:BX -> EBX (unix386.s)
714                             <1>
715                             <1> DSK_WRITE:
```

```
716 000044F4 66B84AC5            <1>       MOV    AX,0C54AH           ; AX = NEC COMMAND, DMA COMMAND
717 000044F8 800D[9E8A0100]80    <1>         OR      byte [MOTOR_STATUS],10000000B ; INDICATE WRITE OPERATION
718 000044FF E848040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
719 00004504 C3                  <1>       RETn
720                              <1>
721                              <1> ;--------------------------------------------------------------------------------
722                              <1> ; DISK_VERF  (AH = 04H)
723                              <1> ;     DISKETTE VERIFY.
724                              <1> ;
725                              <1> ; ON ENTRY:  DI     : DRIVE #
726                              <1> ;            SI-HI  : HEAD #
727                              <1> ;            SI-LOW : # OF SECTORS
728                              <1> ;            ES     : BUFFER SEGMENT
729                              <1> ;            [BP]   : SECTOR #
730                              <1> ;            [BP+1] : TRACK #
731                              <1> ;            [BP+2] : BUFFER OFFSET
732                              <1> ;
733                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
734                              <1> ;--------------------------------------------------------------------------------
735                              <1> DSK_VERF:
736 00004505 8025[9E8A0100]7F    <1>       AND    byte [MOTOR_STATUS],01111111B ; INDICATE A READ OPERATION
737 0000450C 66B842E6            <1>       MOV    AX,0E642H           ; AX = NEC COMMAND, DMA COMMAND
738 00004510 E837040000          <1>       CALL   RD_WR_VF            ; COMMON READ/WRITE/VERIFY
739 00004515 C3                  <1>       RETn
740                              <1>
741                              <1> ;--------------------------------------------------------------------------------
742                              <1> ; DISK_FORMAT      (AH = 05H)
743                              <1> ;     DISKETTE FORMAT.
744                              <1> ;
745                              <1> ; ON ENTRY:  DI     : DRIVE #
746                              <1> ;            SI-HI  : HEAD #
747                              <1> ;            SI-LOW : # OF SECTORS
748                              <1> ;            ES     : BUFFER SEGMENT
749                              <1> ;            [BP]   : SECTOR #
750                              <1> ;            [BP+1] : TRACK #
751                              <1> ;            [BP+2] : BUFFER OFFSET
752                              <1> ;            @DISK_POINTER POINTS TO THE PARAMETER TABLE OF THIS DRIVE
753                              <1> ;
754                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
755                              <1> ;--------------------------------------------------------------------------------
756                              <1> DSK_FORMAT:
757 00004516 E870030000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
758 0000451B E86C050000          <1>       CALL   FMT_INIT            ; ESTABLISH STATE IF UNESTABLISHED
759 00004520 800D[9E8A0100]80    <1>         OR      byte [MOTOR_STATUS], 10000000B ; INDICATE WRITE OPERATION
760 00004527 E8B4050000          <1>       CALL   MED_CHANGE          ; CHECK MEDIA CHANGE AND RESET IF SO
761 0000452C 725D                <1>         JC      short FM_DON             ; MEDIA CHANGED, SKIP
762 0000452E E80F030000          <1>       CALL   SEND_SPEC           ; SEND SPECIFY COMMAND TO NEC
763 00004533 E81A060000          <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMPT RATE IS SAME AS LAST RATE
764 00004538 7405                <1>         JZ      short FM_WR              ; YES, SKIP SPECIFY COMMAND
765 0000453A E8F1050000          <1>       CALL   SEND_RATE           ; SEND DATA RATE TO CONTROLLER
766                              <1> FM_WR:
767 0000453F E8A7060000          <1>       CALL   FMTDMA_SET          ; SET UP THE DMA FOR FORMAT
768 00004544 7245                <1>         JC      short FM_DON             ; RETURN WITH ERROR
769 00004546 B44D                <1>       MOV    AH,04DH                  ; ESTABLISH THE FORMAT COMMAND
770 00004548 E804070000          <1>       CALL   NEC_INIT            ; INITIALIZE THE NEC
771 0000454D 723C                <1>         JC      short FM_DON             ; ERROR - EXIT
772 0000454F B8[8B450000]        <1>         MOV     eAX, FM_DON              ; LOAD ERROR ADDRESS
773 00004554 50                  <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
774 00004555 B203                <1>       MOV    DL,3                ; BYTES/SECTOR VALUE TO NEC
775 00004557 E873090000          <1>       CALL   GET_PARM
776 0000455C E8740A0000          <1>       CALL   NEC_OUTPUT
777 00004561 B204                <1>       MOV    DL,4                ; SECTORS/TRACK VALUE TO NEC
778 00004563 E867090000          <1>       CALL   GET_PARM
779 00004568 E8680A0000          <1>       CALL   NEC_OUTPUT
780 0000456D B207                <1>       MOV    DL,7                ; GAP LENGTH VALUE TO NEC
781 0000456F E85B090000          <1>       CALL   GET_PARM
782 00004574 E85C0A0000          <1>       CALL   NEC_OUTPUT
783 00004579 B208                <1>       MOV    DL,8                ; FILLER BYTE TO NEC
784 0000457B E84F090000          <1>       CALL   GET_PARM
785 00004580 E8500A0000          <1>       CALL   NEC_OUTPUT
786 00004585 58                  <1>       POP    eAX                 ; THROW AWAY ERROR
787 00004586 E844070000          <1>       CALL   NEC_TERM            ; TERMINATE, RECEIVE STATUS, ETC,
788                              <1> FM_DON:
789 0000458B E82C030000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
790 00004590 E866080000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
791 00004595 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
792 00004598 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
793 0000459A C3                  <1>       RETn
794                              <1>
795                              <1> ;--------------------------------------------------------------------------------
796                              <1> ; FNC_ERR
797                              <1> ;     INVALID FUNCTION REQUESTED OR INVALID DRIVE:
798                              <1> ;     SET BAD COMMAND IN STATUS.
799                              <1> ;
800                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
801                              <1> ;--------------------------------------------------------------------------------
802                              <1> FNC_ERR:                           ; INVALID FUNCTION REQUEST
803 0000459B 6689F0              <1>       MOV    AX,SI               ; RESTORE AL
804 0000459E B401                <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
805 000045A0 8825[A08A0100]      <1>       MOV    [DSKETTE_STATUS],AH ; STORE IN DATA AREA
806 000045A6 F9                  <1>       STC                        ; SET CARRY INDICATING ERROR
807 000045A7 C3                  <1>       RETn
808                              <1>
809                              <1> ; 30/08/2020
810                              <1> ; 29/08/2020
811                              <1> ; 01/06/2016
812                              <1> ; 28/05/2016
813                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v.2.0)
814                              <1> ;--------------------------------------------------------------------------------
815                              <1> ; DISK_PARMS (AH = 08H)
816                              <1> ;     READ DRIVE PARAMETERS.
817                              <1> ;
818                              <1> ; ON ENTRY:  DI : DRIVE #
819                              <1> ;            ; 27/05/2016
820                              <1> ;            EBX = Buffer Address for floppy disk parameters table (16 bytes)
```

```
                        <1> ;
                        <1> ; ON EXIT:   CL/[BP]   = BITS 7 & 6 HI 2 BITS OF MAX CYLINDER
                        <1> ;                        BITS 0-5 MAX SECTORS/TRACK
                        <1> ;            CH/[BP+1] = LOW 8 BITS OF MAX CYLINDER
                        <1> ;            BL/[BP+2] = BITS 7-4 = 0
                        <1> ;                        BITS 3-0 = VALID CMOS DRIVE TYPE
                        <1> ;            BH/[BP+3] = 0
                        <1> ;            DL/[BP+4] = # DRIVES INSTALLED (VALUE CHECKED)
                        <1> ;            DH/[BP+5] = MAX HEAD #
                        <1> ;          ** 27/05/2016 - TRDOS 386 (TRDOS v2.0) **
                        <1> ;            ** EBX = Buffer address for floppy disk parameters table **
                        <1> ;            ;DI/[BP+6] = OFFSET TO DISK_BASE
                        <1> ;            ;ES        = SEGMENT OF DISK_BASE
                        <1> ;
                        <1> ;            AX         = 0
                        <1> ;
                        <1> ;            NOTE : THE ABOVE INFORMATION IS STORED IN THE USERS STACK AT
                        <1> ;                   THE LOCATIONS WHERE THE MAIN ROUTINE WILL POP THEM
                        <1> ;                   INTO THE APPROPRIATE REGISTERS BEFORE RETURNING TO THE
                        <1> ;                   CALLER.
                        <1> ;--------------------------------------------------------------------------------
                        <1> DSK_PARMS:
000045A8 E8DE020000     <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH,
                        <1>      ;       MOV    WORD [BP+2],0       ; DRIVE TYPE = 0
                        <1>      ;  MOV     AX, [EQUIP_FLAG]        ; LOAD EQUIPMENT FLAG FOR # DISKETTES
                        <1>      ;  AND     AL,11000001B            ; KEEP DISKETTE DRIVE BITS
                        <1>      ;  MOV     DL,2                    ; DISKETTE DRIVES = 2
                        <1>      ;  CMP     AL,01000001B            ; 2 DRIVES INSTALLED ?
                        <1>      ;  JZ      short STO_DL            ; IF YES JUMP
                        <1>      ;  DEC     DL                      ; DISKETTE DRIVES = 1
                        <1>      ;  CMP     AL,00000001B            ; 1 DRIVE INSTALLED ?
                        <1>      ;  JNZ     short NON_DRV           ; IF NO JUMP
000045AD 29D2           <1>       sub    edx, edx
000045AF 66A1[0E6E0000] <1>       mov    ax, [fd0_type]
000045B5 6621C0         <1>       and    ax, ax
000045B8 0F849B000000   <1>         jz     NON_DRV
000045BE FEC2           <1>       inc    dl
000045C0 20E4           <1>       and    ah, ah
000045C2 7402           <1>       jz     short STO_DL
000045C4 FEC2           <1>       inc    dl
                        <1> STO_DL:
                        <1>       ; 30/08/2020
000045C6 6639FA         <1>       cmp    dx, di
000045C9 0F868A000000   <1>       jna    NON_DRV
                        <1>       ;
                        <1>       ;MOV   [BP+4],DL          ; STORE NUMBER OF DRIVES
000045CF 895508         <1>       mov    [ebp+8], edx ; 20/02/2015
000045D2 6683FF01       <1>       CMP    DI,1               ; CHECK FOR VALID DRIVE
                        <1>       ;JA    short NON_DRV1           ; DRIVE INVALID
000045D6 0F8780000000   <1>       ja     NON_DRV1 ; 29/08/2020
                        <1>       ;MOV   BYTE [BP+5],1      ; MAXIMUM HEAD NUMBER =   1
000045DC C6450901       <1>       mov    byte [ebp+9], 1  ; 20/02/2015
000045E0 E8E1080000     <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN AL
                        <1>       ;;20/02/2015
                        <1>       ;;JC   short CHK_EST      ; IF CMOS BAD CHECKSUM ESTABLISHED
                        <1>       ;;OR   AL,AL              ; TEST FOR NO DRIVE TYPE
000045E5 740F           <1>       JZ     short CHK_EST      ; JUMP IF SO
000045E7 E82B020000     <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
000045EC 7208           <1>       JC     short CHK_EST      ; TYPE NOT IN TABLE (POSSIBLE BAD CMOS)
                        <1>       ;MOV   [BP+2],AL          ; STORE VALID CMOS DRIVE TYPE
                        <1>         ;mov [ebp+4], al ; 06/02/2015
000045EE 8A4B04         <1>       MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
000045F1 8A6B0B         <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
000045F4 EB36           <1>       JMP    SHORT STO_CX       ; CMOS GOOD, USE CMOS
                        <1> CHK_EST:
000045F6 8AA7[AD8A0100] <1>       MOV    AH, [DSK_STATE+eDI] ; LOAD STATE FOR THIS DRIVE
000045FC F6C410         <1>       TEST   AH,MED_DET         ; CHECK FOR ESTABLISHED STATE
000045FF 745B           <1>       JZ     short NON_DRV1           ; CMOS BAD/INVALID OR UNESTABLISHED
                        <1> USE_EST:
00004601 80E4C0         <1>       AND    AH,RATE_MSK        ; ISOLATE STATE
00004604 80FC80         <1>       CMP    AH,RATE_250        ; RATE 250 ?
00004607 757B           <1>       JNE    short USE_EST2           ; NO, GO CHECK OTHER RATE
                        <1>
                        <1> ;-----       DATA RATE IS 250 KBS, TRY 360 KB TABLE FIRST
                        <1>
00004609 B001           <1>       MOV    AL,01              ; DRIVE TYPE 1 (360KB)
0000460B E807020000     <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
00004610 8A4B04         <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
00004613 8A6B0B         <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
00004616 F687[AD8A0100]01 <1>       TEST   byte [DSK_STATE+eDI],TRK_CAPA ; 80 TRACK ?
0000461D 740D           <1>       JZ     short STO_CX       ; MUST BE 360KB DRIVE
                        <1>
                        <1> ;-----       IT IS 1.44 MB DRIVE
                        <1>
                        <1> PARM144:
0000461F B004           <1>       MOV    AL,04              ; DRIVE TYPE 4 (1.44MB)
00004621 E8F1010000     <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
00004626 8A4B04         <1>         MOV     CL, [eBX+MD.SEC_TRK]     ; GET SECTOR/TRACK
00004629 8A6B0B         <1>         MOV     CH, [eBX+MD.MAX_TRK]     ; GET MAX. TRACK NUMBER
                        <1> STO_CX:
0000462C 894D00         <1>       MOV    [eBP],eCX          ; SAVE POINTER IN STACK FOR RETURN
                        <1> ES_DI:
                        <1>       ;MOV   [BP+6],BX          ; ADDRESS OF MEDIA/DRIVE PARM TABLE
                        <1>       ;mov   [ebp+12], ebx ; 06/02/2015
                        <1>       ;MOV   AX,CS              ; SEGMENT MEDIA/DRIVE PARAMETER TABLE
                        <1>       ;MOV   ES,AX              ; ES IS SEGMENT OF TABLE
                        <1>       ;
                        <1>       ; 28/05/2016
                        <1>       ; 27/05/2016
                        <1>       ; return floppy disk parameters table to user
                        <1>       ; in user's buffer, which is pointed by EBX
                        <1>       ;
0000462F 57             <1>       push   edi
00004630 8B7D04         <1>       mov    edi, [ebp+4]             ; ebx (input), user's buffer address
                        <1>       ; 29/08/2020
```

```
00004633 09FF                <1>       or     edi, edi
00004635 7417                <1>       jz     short no_copy_fdpt
                             <1>       ;
00004637 0FB6C0              <1>       movzx  eax, al
0000463A 894504              <1>         mov  [ebp+4], eax   ; ebx      ; drive type (for floppy drives)
                             <1>       ; 01/06/2016 (INT 33h, disk type return for floppy disks, in BL)
0000463D A3[A4960100]        <1>       mov    [user_buffer], eax  ; 01/06/2016 (overwrite ebx return value)
                             <1>       ;(INT 33h, Function 08h will replace user's buffer addr with disk type!)
                             <1>       ;
00004642 89DE                <1>       mov    esi, ebx            ; floppy disk parameter table (16 bytes)
00004644 B910000000          <1>       mov    ecx, 16 ; 16 bytes
00004649 E8F9D40000          <1>         call    transfer_to_user_buffer ; trdosk6.s (16/05/2016)
                             <1> no_copy_fdpt:
0000464E 5F                  <1>       pop    edi
                             <1> DP_OUT:
0000464F E868020000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
00004654 6631C0              <1>       XOR    AX,AX               ; CLEAR
00004657 F8                  <1>       CLC
00004658 C3                  <1>       RETn
                             <1>
                             <1> ;-----       NO DRIYE PRESENT HANDLER
                             <1>
                             <1> NON_DRV:
                             <1>       ;MOV   BYTE [BP+4],0       ; CLEAR NUMBER OF DRIVES
00004659 895508              <1>       mov    [ebp+8], edx ; 0 ; 20/02/2015
                             <1> NON_DRV1:
0000465C 6681FF8000          <1>       CMP    DI,80H              ; CHECK FOR FIXED MEDIA TYPE REQUEST
00004661 720C                <1>       JB     short NON_DRV2            ; CONTINUE IF NOT REQUEST FALL THROUGH
                             <1>
                             <1> ;-----       FIXED DISK REQUEST FALL THROUGH ERROR
                             <1>
00004663 E854020000          <1>       CALL   XLAT_OLD            ; ELSE TRANSLATE TO COMPATIBLE MODE
00004668 6689F0              <1>       MOV    AX,SI               ; RESTORE AL
0000466B B401                <1>       MOV    AH,BAD_CMD          ; SET BAD COMMAND ERROR
0000466D F9                  <1>       STC
0000466E C3                  <1>       RETn
                             <1>
                             <1> NON_DRV2:
                             <1>       ;XOR   AX,AX               ; CLEAR PARMS IF NO DRIVES OR CMOS BAD
0000466F 31C0                <1>       xor    eax, eax
00004671 66894500            <1>       MOV    [eBP],AX            ; TRACKS, SECTORS/TRACK = 0
                             <1>       ;MOV   [BP+5],AH           ; HEAD = 0
00004675 886509              <1>       mov    [ebp+9], ah ; 06/02/2015
                             <1>       ;MOV   [BP+6],AX           ; OFFSET TO DISK_BASE = 0
00004678 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
0000467B E83C020000          <1>       call   XLAT_OLD
                             <1>       ;mov   ah, NOT_RDY ; drive not ready
00004680 B407                <1>       mov    ah, INIT_FAIL ; DRIVE PARAMETER ACTIVITY FAILED
00004682 F9                  <1>       stc    ; cf -> 1, ah = 'drive not ready' error code
00004683 C3                  <1>       retn
                             <1>
                             <1> ;-----       DATA RATE IS EITHER 300 KBS OR 500 KBS, TRY 1.2 MB TABLE FIRST
                             <1>
                             <1> USE_EST2:
00004684 B002                <1>       MOV    AL,02               ; DRIVE TYPE 2 (1.2MB)
00004686 E88C010000          <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
0000468B 8A4B04              <1>         MOV     CL, [eBX+MD.SEC_TRK]    ; GET SECTOR/TRACK
0000468E 8A6B0B              <1>         MOV     CH, [eBX+MD.MAX_TRK]    ; GET MAX. TRACK NUMBER
00004691 80FC40              <1>       CMP    AH,RATE_300         ; RATE 300 ?
00004694 7496                <1>       JZ     short STO_CX        ; MUST BE 1.2MB DRIVE
00004696 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:
00004698 E8EE010000          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
0000469D 8A87[AD8A0100]      <1>       MOV    AL,[DSK_STATE+eDI]  ; GET PRESENT STATE INFORMATION
000046A3 08C0                <1>       OR     AL,AL               ; CHECK FOR NO DRIVE
000046A5 7418                <1>       JZ     short NO_DRV
000046A7 B401                <1>       MOV    AH,NOCHGLN          ; NO CHANGE LINE FOR 40 TRACK DRIVE
000046A9 A801                <1>       TEST   AL,TRK_CAPA         ; IS THIS DRIVE AN 80 TRACK DRIVE?
000046AB 7402                <1>       JZ     short DT_BACK       ; IF NO JUMP
000046AD B402                <1>       MOV    AH,CHGLN            ; CHANGE LINE FOR 80 TRACK DRIVE
                             <1> DT_BACK:
000046AF 6650                <1>       PUSH   AX                  ; SAVE RETURN VALUE
000046B1 E806020000          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
000046B6 6658                <1>       POP    AX                  ; RESTORE RETURN VALUE
000046B8 F8                  <1>       CLC                        ; NO ERROR
000046B9 6689F3              <1>       MOV    BX,SI               ; GET SAVED AL TO BL
000046BC 88D8                <1>       MOV    AL,BL               ; PUT BACK FOR RETURN
000046BE 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
000046BF E8F8010000          <1>       call   XLAT_OLD
000046C4 29C0                <1>       sub    eax, eax
000046C6 F9                  <1>       stc    ; cf = 1 -> drive not ready, ah = 0 (disk type = 0)
000046C7 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:
000046C8 E8BE010000     <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
000046CD 8A87[AD8A0100] <1>       MOV    AL, [DSK_STATE+eDI] ; GET MEDIA STATE INFORMATION
000046D3 08C0           <1>       OR     AL,AL              ; DRIVE PRESENT ?
000046D5 7422           <1>       JZ     short DC_NON       ; JUMP IF NO DRIVE
000046D7 A801           <1>       TEST   AL,TRK_CAPA        ; 80 TRACK DRIVE ?
000046D9 7407           <1>       JZ     short SETIT        ; IF SO , CHECK CHANGE LINE
                        <1> DC0:
000046DB E88D0A0000     <1>         CALL    READ_DSKCHNG            ; GO CHECK STATE OF DISK CHANGE LINE
000046E0 7407           <1>       JZ     short FINIS        ; CHANGE LINE NOT ACTIVE
                        <1>
000046E2 C605[A08A0100]06 <1> SETIT:      MOV    byte [DSKETTE_STATUS], MEDIA_CHANGE ; INDICATE MEDIA REMOVED
                        <1>
000046E9 E8CE010000     <1> FINIS:      CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
000046EE E808070000     <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
000046F3 6689F3         <1>       MOV    BX,SI              ; GET SAVED AL TO BL
000046F6 88D8           <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
000046F8 C3             <1>       RETn
                        <1> DC_NON:
000046F9 800D[A08A0100]80 <1>     OR     byte [DSKETTE_STATUS], TIME_OUT ; SET TIMEOUT, NO DRIVE
00004700 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:
00004702 E884010000     <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
00004707 6656           <1>       PUSH   SI                 ; SAVE DASD TYPE
00004709 6689F0         <1>       MOV    AX,SI              ; AH = ? , AL , DASD TYPE
0000470C 30E4           <1>       XOR    AH,AH              ; AH , 0 , AL , DASD TYPE
0000470E 6689C6         <1>       MOV    SI,AX              ; SI = DASD TYPE
00004711 80A7[AD8A0100]0F <1>     AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
00004718 664E           <1>       DEC    SI                 ; CHECK FOR 320/360K MEDIA & DRIVE
0000471A 7509           <1>       JNZ    short NOT_320      ; BYPASS IF NOT
0000471C 808F[AD8A0100]90 <1>     OR     byte [DSK_STATE+eDI], MED_DET+RATE_250 ; SET TO 320/360
00004723 EB48           <1>       JMP    SHORT S0
                        <1>
                        <1> NOT_320:
00004725 E8B6030000     <1>       CALL   MED_CHANGE         ; CHECK FOR TIME_OUT
0000472A 803D[A08A0100]80 <1>     CMP    byte [DSKETTE_STATUS], TIME_OUT
00004731 743A           <1>       JZ     short S0           ; IF TIME OUT TELL CALLER
                        <1> S3:
00004733 664E           <1>       DEC    SI                 ; CHECK FOR 320/360K IN 1.2M DRIVE
00004735 7509           <1>       JNZ    short NOT_320_12   ; BYPASS IF NOT
00004737 808F[AD8A0100]70 <1>     OR     byte [DSK_STATE+eDI], MED_DET+DBL_STEP+RATE_300 ; SET STATE
0000473E EB2D           <1>       JMP    SHORT S0
                        <1>
                        <1> NOT_320_12:
00004740 664E           <1>       DEC    SI                 ; CHECK FOR 1.2M MEDIA IN 1.2M DRIVE
00004742 7509           <1>       JNZ    short NOT_12       ; BYPASS IF NOT
00004744 808F[AD8A0100]10 <1>     OR     byte [DSK_STATE+eDI], MED_DET+RATE_500 ; SET STATE VARIABLE
0000474B EB20           <1>       JMP    SHORT S0           ; RETURN TO CALLER
                        <1>
                        <1> NOT_12:
0000474D 664E           <1>       DEC    SI                 ; CHECK FOR SET DASD TYPE 04
0000474F 752B           <1>       JNZ    short FS_ERR       ; BAD COMMAND EXIT IF NOT VALID TYPE
                        <1>
00004751 F687[AD8A0100]04 <1>     TEST   byte [DSK_STATE+eDI], DRV_DET ; DRIVE DETERMINED ?
00004758 740B           <1>       JZ     short ASSUME       ; IF STILL NOT DETERMINED ASSUME
0000475A B050           <1>       MOV    AL,MED_DET+RATE_300
0000475C F687[AD8A0100]02 <1>       TEST    byte [DSK_STATE+eDI], FMT_CAPA ; MULTIPLE FORMAT CAPABILITY ?
00004763 7502           <1>       JNZ    short OR_IT_IN            ; IF 1.2 M THEN DATA RATE 300
                        <1>
                        <1> ASSUME:
00004765 B090           <1>       MOV    AL,MED_DET+RATE_250 ; SET UP
                        <1>
                        <1> OR_IT_IN:
00004767 0887[AD8A0100] <1>       OR     [DSK_STATE+eDI], AL ; OR IN THE CORRECT STATE
                        <1> S0:
0000476D E84A010000     <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
00004772 E884060000     <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
00004777 665B           <1>       POP    BX                 ; GET SAVED AL TO BL
00004779 88D8           <1>       MOV    AL,BL              ; PUT BACK FOR RETURN
0000477B C3             <1>       RETn
                        <1>
                        <1> FS_ERR:
0000477C C605[A08A0100]01 <1>     MOV    byte [DSKETTE_STATUS], BAD_CMD ; UNKNOWN STATE,BAD COMMAND
00004783 EBE8           <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 #
```

```
1136                            <1> ;
1137                            <1> ; ON EXIT:
1138                            <1> ;     @DSKETTE_STATUS REFLECTS STATUS
1139                            <1> ;     IF NO ERROR:
1140                            <1> ;           AH = 0
1141                            <1> ;           CY = 0
1142                            <1> ;           ES = SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
1143                            <1> ;           DI/[BP+6] = OFFSET OF MEDIA/DRIVE PARAMETER TABLE
1144                            <1> ;     IF ERROR:
1145                            <1> ;           AH = @DSKETTE_STATUS
1146                            <1> ;           CY = 1
1147                            <1> ;--------------------------------------------------------------------------
1148                            <1> SET_MEDIA:
1149 00004785 E801010000       <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH.
1150 0000478A F687[AD8A0100]01 <1>         TEST    byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR CHANGE LINE AVAILABLE
1151 00004791 7415             <1>       JZ     short SM_CMOS      ; JUMP IF 40 TRACK DRIVE
1152 00004793 E848030000       <1>       CALL   MED_CHANGE         ; RESET CHANGE LINE
1153 00004798 803D[A08A0100]80 <1>       CMP    byte [DSKETTE_STATUS], TIME_OUT ; IF TIME OUT TELL CALLER
1154 0000479F 746B             <1>       JE     short SM_RTN
1155 000047A1 C605[A08A0100]00 <1>       MOV    byte [DSKETTE_STATUS], 0 ; CLEAR STATUS
1156                            <1> SM_CMOS:
1157 000047A8 E819070000       <1>       CALL   CMOS_TYPE          ; RETURN DRIVE TYPE IN (AL)
1158                            <1>       ;;20/02/2015
1159                            <1>       ;;JC   short MD_NOT_FND   ; ERROR IN CMOS
1160                            <1>       ;;OR   AL,AL              ; TEST FOR NO DRIVE
1161 000047AD 745D             <1>       JZ     short SM_RTN       ; RETURN IF SO
1162 000047AF E863000000       <1>       CALL   DR_TYPE_CHECK      ; RTN CS:BX = MEDIA/DRIVE PARAM TBL
1163 000047B4 7231             <1>       JC     short MD_NOT_FND   ; TYPE NOT IN TABLE (BAD CMOS)
1164 000047B6 57               <1>       PUSH   eDI                ; SAVE REG.
1165 000047B7 31DB             <1>       XOR    eBX,eBX                 ; BX = INDEX TO DR. TYPE TABLE
1166 000047B9 B906000000       <1>       MOV    eCX,DR_CNT         ; CX = LOOP COUNT
1167                            <1> DR_SEARCH:
1168 000047BE 8AA3[886D0000]   <1>       MOV    AH, [DR_TYPE+eBX]  ; GET DRIVE TYPE
1169 000047C4 80E47F           <1>       AND    AH,BIT7OFF         ; MASK OUT MSB
1170 000047C7 38E0             <1>       CMP    AL,AH              ; DRIVE TYPE MATCH ?
1171 000047C9 7516             <1>       JNE    short NXT_MD       ; NO, CHECK NEXT DRIVE TYPE
1172                            <1> DR_FND:
1173 000047CB 8BBB[896D0000]   <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAM TABLE
1174                            <1> MD_SEARCH:
1175 000047D1 8A6704           <1>         MOV    AH, [eDI+MD.SEC_TRK]    ; GET SECTOR/TRACK
1176 000047D4 386500           <1>       CMP    [eBP],AH           ; MATCH?
1177 000047D7 7508             <1>       JNE    short NXT_MD       ; NO, CHECK NEXT MEDIA
1178 000047D9 8A670B           <1>         MOV    AH, [eDI+MD.MAX_TRK]    ; GET MAX. TRACK #
1179 000047DC 386501           <1>       CMP    [eBP+1],AH         ; MATCH?
1180 000047DF 740F             <1>       JE     short MD_FND       ; YES, GO GET RATE
1181                            <1> NXT_MD:
1182                            <1>       ;ADD   BX,3               ; CHECK NEXT DRIVE TYPE
1183 000047E1 83C305           <1>         add ebx, 5 ; 18/02/2015
1184 000047E4 E2D8             <1>       LOOP    DR_SEARCH
1185 000047E6 5F               <1>       POP    eDI                ; RESTORE REG.
1186                            <1> MD_NOT_FND:
1187 000047E7 C605[A08A0100]0C <1>       MOV    byte [DSKETTE_STATUS], MED_NOT_FND ; ERROR, MEDIA TYPE NOT FOUND
1188 000047EE EB1C             <1>       JMP    SHORT SM_RTN       ; RETURN
1189                            <1> MD_FND:
1190 000047F0 8A470C           <1>         MOV    AL, [eDI+MD.RATE]       ; GET RATE
1191 000047F3 3C40             <1>       CMP    AL,RATE_300        ; DOUBLE STEP REQUIRED FOR RATE 300
1192 000047F5 7502             <1>       JNE    short MD_SET
1193 000047F7 0C20             <1>       OR     AL,DBL_STEP
1194                            <1> MD_SET:
1195                            <1>       ;MOV   [BP+6],DI          ; SAVE TABLE POINTER IN STACK
1196 000047F9 897D0C           <1>       mov    [ebp+12], edi ; 18/02/2015
1197 000047FC 0C10             <1>       OR     AL,MED_DET         ; SET MEDIA ESTABLISHED
1198 000047FE 5F               <1>       POP    eDI
1199 000047FF 80A7[AD8A0100]0F <1>       AND    byte [DSK_STATE+eDI], ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR STATE
1200 00004806 0887[AD8A0100]   <1>       OR     [DSK_STATE+eDI], AL
1201                            <1>       ;MOV   AX, CS             ; SEGMENT OF MEDIA/DRIVE PARAMETER TABLE
1202                            <1>       ;MOV   ES, AX             ; ES IS SEGMENT OF TABLE
1203                            <1> SM_RTN:
1204 0000480C E8AB000000       <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
1205 00004811 E8E5050000       <1>       CALL   SETUP_END          ; VARIOUS CLEANUPS
1206 00004816 C3               <1>       RETn
1207                            <1>
1208                            <1> ;--------------------------------------------------------------
1209                            <1> ; DR_TYPE_CHECK                                              :
1210                            <1> ;     CHECK IF THE GIVEN DRIVE TYPE IN REGISTER (AL)             :
1211                            <1> ;     IS SUPPORTED IN BIOS DRIVE TYPE TABLE                  :
1212                            <1> ; ON ENTRY:                                          :
1213                            <1> ;     AL = DRIVE TYPE                                        :
1214                            <1> ; ON EXIT:                                           :
1215                            <1> ;     CS = SEGMENT MEDIA/DRIVE PARAMETER TABLE (CODE)            :
1216                            <1> ;     CY = 0       DRIVE TYPE SUPPORTED                          :
1217                            <1> ;          BX = OFFSET TO MEDIA/DRIVE PARAMETER TABLE           :
1218                            <1> ;     CY = 1 DRIVE TYPE NOT SUPPORTED                :
1219                            <1> ; REGISTERS ALTERED: eBX                                     :
1220                            <1> ;--------------------------------------------------------------
1221                            <1> DR_TYPE_CHECK:
1222 00004817 6650             <1>       PUSH   AX
1223 00004819 51               <1>       PUSH   eCX
1224 0000481A 31DB             <1>       XOR    eBX,eBX                  ; BX = INDEX TO DR_TYPE TABLE
1225 0000481C B906000000       <1>       MOV    eCX,DR_CNT         ; CX = LOOP COUNT
1226                            <1> TYPE_CHK:
1227 00004821 8AA3[886D0000]   <1>       MOV    AH,[DR_TYPE+eBX]   ; GET DRIVE TYPE
1228 00004827 38E0             <1>       CMP    AL,AH              ; DRIVE TYPE MATCH?
1229 00004829 740D             <1>       JE     short DR_TYPE_VALID ; YES, RETURN WITH CARRY RESET
1230                            <1>       ;ADD   BX,3               ; CHECK NEXT DRIVE TYPE
1231 0000482B 83C305           <1>         add ebx, 5 ; 16/02/2015 (32 bit address modification)
1232 0000482E E2F1             <1>       LOOP    TYPE_CHK
1233                            <1>       ;
1234 00004830 BB[E76D0000]     <1>       mov    ebx, MD_TBL6       ; 1.44MB fd parameter table
1235                            <1>                                 ; Default for GET_PARM (11/12/2014)
1236                            <1>       ;
1237 00004835 F9               <1>       STC                       ; DRIVE TYPE NOT FOUND IN TABLE
1238 00004836 EB06             <1>       JMP    SHORT TYPE_RTN
1239                            <1> DR_TYPE_VALID:
1240 00004838 8B9B[896D0000]   <1>       MOV    eBX,[DR_TYPE+eBX+1]       ; BX = MEDIA TABLE
```

```
                        <1> TYPE_RTN:
0000483E 59             <1>       POP    eCX
0000483F 6658           <1>       POP    AX
00004841 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------
                        <1> ; SEND_SPEC                                                  :
                        <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
                        <1> ;     THE DRIVE PARAMETER TABLE POINTED BY @DISK_POINTER   :
                        <1> ; ON ENTRY:  @DISK_POINTER = DRIVE PARAMETER TABLE         :
                        <1> ; ON EXIT:   NONE                                          :
                        <1> ; REGISTERS ALTERED: CX, DX                                :
                        <1> ;--------------------------------------------------------------
                        <1> SEND_SPEC:
00004842 50             <1>       PUSH   eAX                ; SAVE AX
00004843 B8[69480000]   <1>       MOV    eAX, SPECBAC       ; LOAD ERROR ADDRESS
00004848 50             <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
00004849 B403           <1>       MOV    AH,03H             ; SPECIFY COMMAND
0000484B E885070000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
00004850 28D2           <1>       SUB    DL,DL              ; FIRST SPECIFY BYTE
00004852 E878060000     <1>       CALL   GET_PARM           ; GET PARAMETER TO AH
00004857 E879070000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
0000485C B201           <1>       MOV    DL,1               ; SECOND SPECIFY BYTE
0000485E E86C060000     <1>       CALL   GET_PARM           ; GET PARAMETER TO AH
00004863 E86D070000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
00004868 58             <1>       POP    eAX                ; POP ERROR RETURN
                        <1> SPECBAC:
00004869 58             <1>       POP    eAX                ; RESTORE ORIGINAL AX VALUE
0000486A C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------
                        <1> ; SEND_SPEC_MD                                               :
                        <1> ;     SEND THE SPECIFY COMMAND TO CONTROLLER USING DATA FROM     :
                        <1> ;     THE MEDIA/DRIVE PARAMETER TABLE POINTED BY (CS:BX)   :
                        <1> ; ON ENTRY:  CS:BX = MEDIA/DRIVE PARAMETER TABLE           :
                        <1> ; ON EXIT:   NONE                                          :
                        <1> ; REGISTERS ALTERED: AX                                    :
                        <1> ;--------------------------------------------------------------
                        <1> SEND_SPEC_MD:
0000486B 50             <1>       PUSH   eAX                ; SAVE RATE DATA
0000486C B8[89480000]   <1>       MOV    eAX, SPEC_ESBAC          ; LOAD ERROR ADDRESS
00004871 50             <1>       PUSH   eAX                ; PUSH NEC_OUT ERROR RETURN
00004872 B403           <1>       MOV    AH,03H             ; SPECIFY COMMAND
00004874 E85C070000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
00004879 8A23           <1>         MOV     AH, [eBX+MD.SPEC1]      ; GET 1ST SPECIFY BYTE
0000487B E855070000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
00004880 8A6301         <1>         MOV     AH, [eBX+MD.SPEC2]      ; GET SECOND SPECIFY BYTE
00004883 E84D070000     <1>       CALL   NEC_OUTPUT         ; OUTPUT THE COMMAND
00004888 58             <1>       POP    eAX                ; POP ERROR RETURN
                        <1> SPEC_ESBAC:
00004889 58             <1>       POP    eAX                ; RESTORE ORIGINAL AX VALUE
0000488A C3             <1>       RETn
                        <1>
                        <1> ;-----------------------------------------------------------------------------
                        <1> ; XLAT_NEW
                        <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM COMPATIBLE
                        <1> ;     MODE TO NEW ARCHITECTURE.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE #
                        <1> ;-----------------------------------------------------------------------------
                        <1> XLAT_NEW:
0000488B 83FF01         <1>       CMP    eDI,1                    ; VALID DRIVE
0000488E 7725           <1>       JA     short XN_OUT             ; IF INVALID BACK
00004890 80BF[AD8A0100]00 <1>     CMP    byte [DSK_STATE+eDI], 0          ; NO DRIVE ?
00004897 741D           <1>       JZ     short DO_DET             ; IF NO DRIVE ATTEMPT DETERMINE
00004899 6689F9         <1>       MOV    CX,DI                    ; CX = DRIVE NUMBER
0000489C C0E102         <1>       SHL    CL,2                     ; CL = SHIFT COUNT, A=0, B=4
0000489F A0[AC8A0100]   <1>       MOV    AL, [HF_CNTRL]                ; DRIVE INFORMATION
000048A4 D2C8           <1>       ROR    AL,CL                    ; TO LOW NIBBLE
000048A6 2407           <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA     ; KEEP DRIVE BITS
000048A8 80A7[AD8A0100]F8 <1>       AND     byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA)
000048AF 0887[AD8A0100] <1>       OR     [DSK_STATE+eDI], AL       ; UPDATE DRIVE STATE
                        <1> XN_OUT:
000048B5 C3             <1>       RETn
                        <1> DO_DET:
000048B6 E8BF080000     <1>       CALL   DRIVE_DET                ; TRY TO DETERMINE
000048BB C3             <1>       RETn
                        <1>
                        <1> ;-----------------------------------------------------------------------------
                        <1> ; XLAT_OLD
                        <1> ;     TRANSLATES DISKETTE STATE LOCATIONS FROM NEW
                        <1> ;     ARCHITECTURE TO COMPATIBLE MODE.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE
                        <1> ;-----------------------------------------------------------------------------
                        <1> XLAT_OLD:
000048BC 83FF01         <1>       CMP    eDI,1               ; VALID DRIVE ?
                        <1>         ;JA     short XO_OUT            ; IF INVALID BACK
000048BF 0F8786000000   <1>         ja      XO_OUT
000048C5 80BF[AD8A0100]00 <1>       CMP  byte [DSK_STATE+eDI],0    ; NO DRIVE ?
000048CC 747D           <1>       JZ     short XO_OUT        ; IF NO DRIVE TRANSLATE DONE
                        <1>
                        <1> ;-----      TEST FOR SAVED DRIVE INFORMATION ALREADY SET
                        <1>
000048CE 6689F9         <1>       MOV    CX,DI               ; CX = DRIVE NUMBER
000048D1 C0E102         <1>       SHL    CL,2                ; CL = SHIFT COUNT, A=0, B=4
000048D4 B402           <1>       MOV    AH,FMT_CAPA         ; LOAD MULTIPLE DATA RATE BIT MASK
000048D6 D2CC           <1>       ROR    AH,CL               ; ROTATE BY MASK
000048D8 8425[AC8A0100] <1>       TEST   [HF_CNTRL], AH            ; MULTIPLE-DATA RATE DETERMINED ?
000048DE 751C           <1>       JNZ    short SAVE_SET            ; IF SO, NO NEED TO RE-SAVE
                        <1>
                        <1> ;-----      ERASE DRIVE BITS IN @HF_CNTRL FOR THIS DRIVE
                        <1>
000048E0 B407           <1>       MOV    AH,DRV_DET+FMT_CAPA+TRK_CAPA ; MASK TO KEEP
000048E2 D2CC           <1>       ROR    AH,CL               ; FIX MASK TO KEEP
```

```
1346 000048E4 F6D4                <1>       NOT    AH                  ; TRANSLATE MASK
1347 000048E6 2025[AC8A0100]      <1>       AND    [HF_CNTRL], AH            ; KEEP BITS FROM OTHER DRIVE INTACT
1348                              <1>
1349                              <1> ;-----      ACCESS CURRENT DRIVE BITS AND STORE IN @HF_CNTRL
1350                              <1>
1351 000048EC 8A87[AD8A0100]      <1>       MOV    AL, [DSK_STATE+eDI] ; ACCESS STATE
1352 000048F2 2407                <1>       AND    AL,DRV_DET+FMT_CAPA+TRK_CAPA ; KEEP DRIVE BITS
1353 000048F4 D2C8                <1>       ROR    AL,CL               ; FIX FOR THIS DRIVE
1354 000048F6 0805[AC8A0100]      <1>       OR     [HF_CNTRL], AL            ; UPDATE SAVED DRIVE STATE
1355                              <1>
1356                              <1> ;-----      TRANSLATE TO COMPATIBILITY MODE
1357                              <1>
1358                              <1> SAVE_SET:
1359 000048FC 8AA7[AD8A0100]      <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
1360 00004902 88E7                <1>       MOV    BH,AH               ; TO BH FOR LATER
1361 00004904 80E4C0              <1>       AND    AH,RATE_MSK         ; KEEP ONLY RATE
1362 00004907 80FC00              <1>       CMP    AH,RATE_500         ; RATE 500 ?
1363 0000490A 7410                <1>       JZ     short CHK_144       ; YES 1.2/1.2 OR 1.44/1.44
1364 0000490C B001                <1>       MOV    AL,M3D1U            ; AL = 360 IN 1.2 UNESTABLISHED
1365 0000490E 80FC40              <1>       CMP    AH,RATE_300         ; RATE 300 ?
1366 00004911 7518                <1>       JNZ    short CHK_250       ; NO, 360/360, 720/720 OR 720/1.44
1367 00004913 F6C720              <1>       TEST   BH,DBL_STEP         ; CHECK FOR DOUBLE STEP
1368 00004916 751F                <1>       JNZ    short TST_DET       ; MUST BE 360 IN 1.2
1369                              <1> UNKNO:
1370 00004918 B007                <1>       MOV    AL,MED_UNK          ; NONE OF THE ABOVE
1371 0000491A EB22                <1>       JMP    SHORT AL_SET        ; PROCESS COMPLETE
1372                              <1> CHK_144:
1373 0000491C E8A5050000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
1374                              <1>       ;;20/02/2015
1375                              <1>       ;;JC   short UNKNO         ; ERROR, SET 'NONE OF ABOVE'
1376 00004921 74F5                <1>       jz     short UNKNO ;; 20/02/2015
1377 00004923 3C02                <1>       CMP    AL,2                ; 1.2MB DRIVE ?
1378 00004925 75F1                <1>       JNE    short UNKNO         ; NO, GO SET 'NONE OF ABOVE'
1379 00004927 B002                <1>       MOV    AL,M1D1U            ; AL = 1.2 IN 1.2 UNESTABLISHED
1380 00004929 EB0C                <1>       JMP    SHORT TST_DET
1381                              <1> CHK_250:
1382 0000492B B000                <1>       MOV    AL,M3D3U            ; AL = 360 IN 360 UNESTABLISHED
1383 0000492D 80FC80              <1>       CMP    AH,RATE_250         ; RATE 250 ?
1384 00004930 75E6                <1>       JNZ    short UNKNO         ; IF SO FALL IHRU
1385 00004932 F6C701              <1>       TEST   BH,TRK_CAPA         ; 80 TRACK CAPABILITY ?
1386 00004935 75E1                <1>       JNZ    short UNKNO         ; IF SO JUMP, FALL THRU TEST DET
1387                              <1> TST_DET:
1388 00004937 F6C710              <1>       TEST   BH,MED_DET          ; DETERMINED ?
1389 0000493A 7402                <1>       JZ     short AL_SET        ; IF NOT THEN SET
1390 0000493C 0403                <1>       ADD    AL,3                ; MAKE DETERMINED/ESTABLISHED
1391                              <1> AL_SET:
1392 0000493E 80A7[AD8A0100]F8    <1>       AND    byte [DSK_STATE+eDI], ~(DRV_DET+FMT_CAPA+TRK_CAPA) ; CLEAR DRIVE
1393 00004945 0887[AD8A0100]      <1>       OR     [DSK_STATE+eDI], AL ; REPLACE WITH COMPATIBLE MODE
1394                              <1> XO_OUT:
1395 0000494B C3                  <1>       RETn
1396                              <1>
1397                              <1> ;--------------------------------------------------------------------------
1398                              <1> ; RD_WR_VF
1399                              <1> ;     COMMON READ, WRITE AND VERIFY:
1400                              <1> ;     MAIN LOOP FOR STATE RETRIES.
1401                              <1> ;
1402                              <1> ; ON ENTRY:  AH = READ/WRITE/VERIFY NEC PARAMETER
1403                              <1> ;            AL = READ/WRITE/VERIFY DMA PARAMETER
1404                              <1> ;
1405                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
1406                              <1> ;--------------------------------------------------------------------------
1407                              <1> RD_WR_VF:
1408 0000494C 6650                <1>       PUSH   AX                  ; SAVE DMA, NEC PARAMETERS
1409 0000494E E838FFFFFF          <1>       CALL   XLAT_NEW            ; TRANSLATE STATE TO PRESENT ARCH.
1410 00004953 E8F3000000          <1>       CALL   SETUP_STATE         ; INITIALIZE START AND END RATE
1411 00004958 6658                <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY
1412                              <1> DO_AGAIN:
1413 0000495A 6650                <1>       PUSH   AX                  ; SAVE READ/WRITE/VERIFY PARAMETER
1414 0000495C E87F010000          <1>       CALL   MED_CHANGE          ; MEDIA CHANGE AND RESET IF CHANGED
1415 00004961 6658                <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY
1416 00004963 0F82C9000000        <1>         JC     RWV_END                  ; MEDIA CHANGE ERROR OR TIME-OUT
1417                              <1> RWV:
1418 00004969 6650                <1>       PUSH   AX                  ; SAVE READ/WRITE/VERIFY PARAMETER
1419 0000496B 8AB7[AD8A0100]      <1>       MOV    DH, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
1420 00004971 80E6C0              <1>       AND    DH,RATE_MSK         ; KEEP ONLY RATE
1421 00004974 E84D050000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL (AL)
1422                              <1>       ;;20/02/2015
1423                              <1>       ;;JC   short RWV_ASSUME    ; ERROR IN CMOS
1424 00004979 7451                <1>       jz     short RWV_ASSUME ; 20/02/2015
1425 0000497B 3C01                <1>       CMP    AL,1                ; 40 TRACK DRIVE?
1426 0000497D 750D                <1>       JNE    short RWV_1         ; NO, BYPASS CMOS VALIDITY CHECK
1427 0000497F F687[AD8A0100]01    <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; CHECK FOR 40 TRACK DRIVE
1428 00004986 7413                <1>       JZ     short RWV_2         ; YES, CMOS IS CORRECT
1429 00004988 B002                <1>       MOV    AL,2                ; CHANGE TO 1.2M
1430 0000498A EB0F                <1>       JMP    SHORT RWV_2
1431                              <1> RWV_1:
1432 0000498C 720D                <1>       JB     short RWV_2         ; NO DRIVE SPECIFIED, CONTINUE
1433 0000498E F687[AD8A0100]01    <1>       TEST    byte [DSK_STATE+eDI], TRK_CAPA ; IS IT REALLY 40 TRACK?
1434 00004995 7504                <1>       JNZ    short RWV_2         ; NO, 80 TRACK
1435 00004997 B001                <1>       MOV    AL,1                ; IT IS 40 TRACK, FIX CMOS VALUE
1436 00004999 EB04                <1>       jmp    short rwv_3
1437                              <1> RWV_2:
1438 0000499B 08C0                <1>       OR     AL,AL               ; TEST FOR NO DRIVE
1439 0000499D 742D                <1>       JZ     short RWV_ASSUME    ; ASSUME TYPE, USE MAX TRACK
1440                              <1> rwv_3:
1441 0000499F E873FEFFFF          <1>       CALL   DR_TYPE_CHECK       ; RTN CS:BX = MEDIA/DRIVE PARAM TBL.
1442 000049A4 7226                <1>       JC     short RWV_ASSUME    ; TYPE NOT IN TABLE (BAD CMOS)
1443                              <1>
1444                              <1> ;-----      SEARCH FOR MEDIA/DRIVE PARAMETER TABLE
1445                              <1>
1446 000049A6 57                  <1>       PUSH   eDI                 ; SAVE DRIVE #
1447 000049A7 31DB                <1>       XOR    eBX,eBX                   ; BX = INDEX TO DR_TYPE TABLE
1448 000049A9 B906000000          <1>       MOV    eCX,DR_CNT          ; CX = LOOP COUNT
1449                              <1> RWV_DR_SEARCH:
1450 000049AE 8AA3[886D0000]      <1>       MOV    AH, [DR_TYPE+eBX]   ; GET DRIVE TYPE
```

```
1451 000049B4 80E47F              <1>       AND    AH,BIT7OFF          ; MASK OUT MSB
1452 000049B7 38E0                <1>       CMP    AL,AH               ; DRIVE TYPE MATCH?
1453 000049B9 750B                <1>       JNE    short RWV_NXT_MD    ; NO, CHECK NEXT DRIVE TYPE
1454                              <1> RWV_DR_FND:
1455 000049BB 8BBB[896D0000]      <1>       MOV    eDI, [DR_TYPE+eBX+1]      ; DI = MEDIA/DRIVE PARAMETER TABLE
1456                              <1> RWV_MD_SEARH:
1457 000049C1 3A770C              <1>         CMP     DH, [eDI+MD.RATE]       ; MATCH?
1458 000049C4 741B                <1>       JE     short RWV_MD_FND    ; YES, GO GET 1ST SPECIFY BYTE
1459                              <1> RWV_NXT_MD:
1460                              <1>       ;ADD   BX,3                ; CHECK NEXT DRIVE TYPE
1461 000049C6 83C305              <1>       add    eBX, 5
1462 000049C9 E2E3                <1>       LOOP   RWV_DR_SEARCH
1463 000049CB 5F                  <1>       POP    eDI                 ; RESTORE DRIVE #
1464                              <1>
1465                              <1> ;-----      ASSUME PRIMARY DRIVE IS INSTALLED AS SHIPPED
1466                              <1>
1467                              <1> RWV_ASSUME:
1468 000049CC BB[A66D0000]        <1>       MOV    eBX, MD_TBL1        ; POINT TO 40 TRACK 250 KBS
1469 000049D1 F687[AD8A0100]01    <1>       TEST   byte [DSK_STATE+eDI], TRK_CAPA ; TEST FOR 80 TRACK
1470 000049D8 740A                <1>       JZ     short RWV_MD_FND1   ; MUST BE 40 TRACK
1471 000049DA BB[C06D0000]        <1>       MOV    eBX, MD_TBL3        ; POINT TO 80 TRACK 500 KBS
1472 000049DF EB03                <1>       JMP    short RWV_MD_FND1   ; GO SPECIFY PARAMTERS
1473                              <1>
1474                              <1> ;-----      CS:BX POINTS TO MEDIA/DRIVE PARAMETER TABLE
1475                              <1>
1476                              <1> RWV_MD_FND:
1477 000049E1 89FB                <1>       MOV    eBX,eDI                   ; BX = MEDIA/DRIVE PARAMETER TABLE
1478 000049E3 5F                  <1>       POP    eDI                 ; RESTORE DRIVE #
1479                              <1>
1480                              <1> ;-----      SEND THE SPECIFY COMMAND TO THE CONTROLLER
1481                              <1>
1482                              <1> RWV_MD_FND1:
1483 000049E4 E882FEFFFF          <1>       CALL   SEND_SPEC_MD
1484 000049E9 E864010000          <1>       CALL   CHK_LASTRATE        ; ZF=1 ATTEMP RATE IS SAME AS LAST RATE
1485 000049EE 7405                <1>       JZ     short RWV_DBL       ; YES,SKIP SEND RATE COMMAND
1486 000049F0 E83B010000          <1>       CALL   SEND_RATE           ; SEND DATA RATE TO NEC
1487                              <1> RWV_DBL:
1488 000049F5 53                  <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
1489 000049F6 E822040000          <1>       CALL   SETUP_DBL           ; CHECK FOR DOUBLE STEP
1490 000049FB 5B                  <1>       POP    eBX                 ; RESTORE ADDRESS
1491 000049FC 7226                <1>       JC     short CHK_RET       ; ERROR FROM READ ID, POSSIBLE RETRY
1492 000049FE 6658                <1>       POP    AX                  ; RESTORE NEC, DMA COMMAND
1493 00004A00 6650                <1>       PUSH   AX                  ; SAVE NEC COMMAND
1494 00004A02 53                  <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
1495 00004A03 E861010000          <1>       CALL   DMA_SETUP           ; SET UP THE DMA
1496 00004A08 5B                  <1>       POP    eBX
1497 00004A09 6658                <1>       POP    AX                  ; RESTORE NEC COMMAND
1498 00004A0B 722F                <1>       JC     short RWV_BAC       ; CHECK FOR DMA BOUNDARY ERROR
1499 00004A0D 6650                <1>       PUSH   AX                  ; SAVE NEC COMMAND
1500 00004A0F 53                  <1>       PUSH   eBX                 ; SAVE MEDIA/DRIVE PARAM TBL ADDRESS
1501 00004A10 E83C020000          <1>       CALL   NEC_INIT            ; INITIALIZE NEC
1502 00004A15 5B                  <1>       POP    eBX                 ; RESTORE ADDRESS
1503 00004A16 720C                <1>       JC     short CHK_RET       ; ERROR - EXIT
1504 00004A18 E866020000          <1>       CALL   RWV_COM                  ; OP CODE COMMON TO READ/WRITE/VERIFY
1505 00004A1D 7205                <1>       JC     short CHK_RET       ; ERROR - EXIT
1506 00004A1F E8AB020000          <1>       CALL   NEC_TERM            ; TERMINATE, GET STATUS, ETC.
1507                              <1> CHK_RET:
1508 00004A24 E84A030000          <1>       CALL   RETRY               ; CHECK FOR, SETUP RETRY
1509 00004A29 6658                <1>       POP    AX                  ; RESTORE READ/WRITE/VERIFY PARAMETER
1510 00004A2B 7305                <1>       JNC    short RWV_END       ; CY = 0 NO RETRY
1511 00004A2D E928FFFFFF          <1>         JMP     DO_AGAIN               ; CY = 1 MEANS RETRY
1512                              <1> RWV_END:
1513 00004A32 E8F4020000          <1>       CALL   DSTATE              ; ESTABLISH STATE IF SUCCESSFUL
1514 00004A37 E887030000          <1>       CALL   NUM_TRANS           ; AL = NUMBER TRANSFERRED
1515                              <1> RWV_BAC:                         ; BAD DMA ERROR ENTRY
1516 00004A3C 6650                <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
1517 00004A3E E879FEFFFF          <1>       CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
1518 00004A43 6658                <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
1519 00004A45 E8B1030000          <1>       CALL   SETUP_END           ; VARIOUS CLEANUPS
1520 00004A4A C3                  <1>       RETn
1521                              <1>
1522                              <1> ;--------------------------------------------------------------------------
1523                              <1> ; SETUP_STATE:     INITIALIZES START AND END RATES.
1524                              <1> ;--------------------------------------------------------------------------
1525                              <1> SETUP_STATE:
1526 00004A4B F687[AD8A0100]10    <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; MEDIA DETERMINED ?
1527 00004A52 7537                <1>       JNZ    short J1C           ; NO STATES IF DETERMINED
1528 00004A54 66B84000            <1>         MOV     AX,(RATE_500*256)+RATE_300  ; AH = START RATE, AL = END RATE
1529 00004A58 F687[AD8A0100]04    <1>       TEST   byte [DSK_STATE+eDI],DRV_DET ; DRIVE ?
1530 00004A5F 740D                <1>       JZ     short AX_SET        ; DO NOT KNOW DRIVE
1531 00004A61 F687[AD8A0100]02    <1>       TEST   byte [DSK_STATE+eDI], FMT_CAPA ; MULTI-RATE?
1532 00004A68 7504                <1>       JNZ    short AX_SET        ; JUMP IF YES
1533 00004A6A 66B88080            <1>         MOV     AX,RATE_250*257         ; START A END RATE 250 FOR 360 DRIVE
1534                              <1> AX_SET:
1535 00004A6E 80A7[AD8A0100]1F    <1>       AND    byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP) ; TURN OFF THE RATE
1536 00004A75 08A7[AD8A0100]      <1>       OR     [DSK_STATE+eDI], AH ; RATE FIRST TO TRY
1537 00004A7B 8025[A88A0100]F3    <1>       AND    byte [LASTRATE], ~STRT_MSK ; ERASE LAST TO TRY RATE BITS
1538 00004A82 C0C804              <1>       ROR    AL,4                ; TO OPERATION LAST RATE LOCATION
1539 00004A85 0805[A88A0100]      <1>       OR     [LASTRATE], AL            ; LAST RATE
1540                              <1> J1C:
1541 00004A8B C3                  <1>       RETn
1542                              <1>
1543                              <1> ;--------------------------------------------------------------------------
1544                              <1> ;  FMT_INIT: ESTABLISH STATE IF UNESTABLISHED AT FORMAT TIME.
1545                              <1> ;--------------------------------------------------------------------------
1546                              <1> FMT_INIT:
1547 00004A8C F687[AD8A0100]10    <1>       TEST   byte [DSK_STATE+eDI], MED_DET ; IS MEDIA ESTABLISHED
1548 00004A93 7546                <1>       JNZ    short F1_OUT        ; IF SO RETURN
1549 00004A95 E82C040000          <1>       CALL   CMOS_TYPE           ; RETURN DRIVE TYPE IN AL
1550                              <1>       ;; 20/02/2015
1551                              <1>       ;;JC   short CL_DRV        ; ERROR IN CMOS ASSUME NO DRIVE
1552 00004A9A 7440                <1>       jz     short CL_DRV ;; 20/02/2015
1553 00004A9C FEC8                <1>       DEC    AL                  ; MAKE ZERO ORIGIN
1554                              <1>       ;;JS   short CL_DRV        ; NO DRIVE IF AL 0
1555 00004A9E 8AA7[AD8A0100]      <1>       MOV    AH, [DSK_STATE+eDI] ; AH = CURRENT STATE
```

```
1556 00004AA4 80E40F              <1>       AND   AH, ~(MED_DET+DBL_STEP+RATE_MSK) ; CLEAR
1557 00004AA7 08C0                <1>       OR    AL,AL               ; CHECK FOR 360
1558 00004AA9 7505                <1>       JNZ   short N_360         ; IF 360 WILL BE 0
1559 00004AAB 80CC90              <1>       OR    AH,MED_DET+RATE_250 ; ESTABLISH MEDIA
1560 00004AAE EB25                <1>       JMP   SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
1561                              <1> N_360:
1562 00004AB0 FEC8                <1>       DEC   AL                  ; 1.2 M DRIVE
1563 00004AB2 7505                <1>       JNZ   short N_12          ; JUMP IF NOT
1564                              <1> F1_RATE:
1565 00004AB4 80CC10              <1>       OR    AH,MED_DET+RATE_500 ; SET FORMAT RATE
1566 00004AB7 EB1C                <1>       JMP   SHORT SKP_STATE           ; SKIP OTHER STATE PROCESSING
1567                              <1> N_12:
1568 00004AB9 FEC8                <1>       DEC   AL                  ; CHECK FOR TYPE 3
1569 00004ABB 750F                <1>       JNZ   short N_720         ; JUMP IF NOT
1570 00004ABD F6C404              <1>       TEST  AH,DRV_DET          ; IS DRIVE DETERMINED
1571 00004AC0 7410                <1>       JZ    short ISNT_12       ; TREAT AS NON 1.2 DRIVE
1572 00004AC2 F6C402              <1>       TEST  AH,FMT_CAPA         ; IS 1.2M
1573 00004AC5 740B                <1>       JZ    short ISNT_12       ; JUMP IF NOT
1574 00004AC7 80CC50              <1>       OR    AH,MED_DET+RATE_300 ; RATE 300
1575 00004ACA EB09                <1>       JMP   SHORT SKP_STATE           ; CONTINUE
1576                              <1> N_720:
1577 00004ACC FEC8                <1>       DEC   AL                  ; CHECK FOR TYPE 4
1578 00004ACE 750C                <1>       JNZ   short CL_DRV        ; NO DRIVE, CMOS BAD
1579 00004AD0 EBE2                <1>       JMP   SHORT F1_RATE
1580                              <1> ISNT_12:
1581 00004AD2 80CC90              <1>       OR    AH,MED_DET+RATE_250 ; MUST BE RATE 250
1582                              <1>
1583                              <1> SKP_STATE:
1584 00004AD5 88A7[AD8A0100]      <1>       MOV   [DSK_STATE+eDI], AH ; STORE AWAY
1585                              <1> F1_OUT:
1586 00004ADB C3                  <1>       RETn
1587                              <1> CL_DRV:
1588 00004ADC 30E4                <1>       XOR   AH,AH               ; CLEAR STATE
1589 00004ADE EBF5                <1>       JMP   SHORT SKP_STATE           ; SAVE IT
1590                              <1>
1591                              <1> ;--------------------------------------------------------------------------
1592                              <1> ; MED_CHANGE
1593                              <1> ;     CHECKS FOR MEDIA CHANGE, RESETS MEDIA CHANGE,
1594                              <1> ;     CHECKS MEDIA CHANGE AGAIN.
1595                              <1> ;
1596                              <1> ; ON EXIT:  CY = 1 MEANS MEDIA CHANGE OR TIMEOUT
1597                              <1> ;           @DSKETTE_STATUS = ERROR CODE
1598                              <1> ;--------------------------------------------------------------------------
1599                              <1> MED_CHANGE:
1600 00004AE0 E888060000          <1>       CALL  READ_DSKCHNG        ; READ DISK CHANCE LINE STATE
1601 00004AE5 7447                <1>       JZ    short MC_OUT        ; BYPASS HANDLING DISK CHANGE LINE
1602 00004AE7 80A7[AD8A0100]EF    <1>       AND   byte [DSK_STATE+eDI], ~MED_DET ; CLEAR STATE FOR THIS DRIVE
1603                              <1>
1604                              <1> ;     THIS SEQUENCE ENSURES WHENEVER A DISKETTE IS CHANGED THAT
1605                              <1> ;     ON THE NEXT OPERATION THE REQUIRED MOTOR START UP TIME WILL
1606                              <1> ;     BE WAITED. (DRIVE MOTOR MAY GO OFF UPON DOOR OPENING).
1607                              <1>
1608 00004AEE 6689F9              <1>       MOV   CX,DI               ; CL = DRIVE 0
1609 00004AF1 B001                <1>       MOV   AL,1                ; MOTOR ON BIT MASK
1610 00004AF3 D2E0                <1>       SHL   AL,CL               ; TO APPROPRIATE POSITION
1611 00004AF5 F6D0                <1>       NOT   AL                  ; KEEP ALL BUT MOTOR ON
1612 00004AF7 FA                  <1>       CLI                       ; NO INTERRUPTS
1613 00004AF8 2005[9E8A0100]      <1>       AND   [MOTOR_STATUS], AL  ; TURN MOTOR OFF INDICATOR
1614 00004AFE FB                  <1>       STI                       ; INTERRUPTS ENABLED
1615 00004AFF E810040000          <1>       CALL  MOTOR_ON            ; TURN MOTOR ON
1616                              <1>
1617                              <1> ;-----      THIS SEQUENCE OF SEEKS IS USED TO RESET DISKETTE CHANGE SIGNAL
1618                              <1>
1619 00004B04 E850F9FFFF          <1>       CALL  DSK_RESET           ; RESET NEC
1620 00004B09 B501                <1>       MOV   CH,01H              ; MOVE TO CYLINDER 1
1621 00004B0B E8FF040000          <1>       CALL  SEEK                ; ISSUE SEEK
1622 00004B10 30ED                <1>       XOR   CH,CH               ; MOVE TO CYLINDER 0
1623 00004B12 E8F8040000          <1>       CALL  SEEK                ; ISSUE SEEK
1624 00004B17 C605[A08A0100]06    <1>       MOV   byte [DSKETTE_STATUS], MEDIA_CHANGE ; STORE IN STATUS
1625                              <1> OK1:
1626 00004B1E E84A060000          <1>       CALL  READ_DSKCHNG        ; CHECK MEDIA CHANGED AGAIN
1627 00004B23 7407                <1>       JZ    short OK2           ; IF ACTIVE, NO DISKETTE, TIMEOUT
1628                              <1> OK4:
1629 00004B25 C605[A08A0100]80    <1>       MOV   byte [DSKETTE_STATUS], TIME_OUT ; TIMEOUT IF DRIVE EMPTY
1630                              <1> OK2:
1631 00004B2C F9                  <1>       STC                       ; MEDIA CHANGED, SET CY
1632 00004B2D C3                  <1>       RETn
1633                              <1> MC_OUT:
1634 00004B2E F8                  <1>       CLC                       ; NO MEDIA CHANGED, CLEAR CY
1635 00004B2F C3                  <1>       RETn
1636                              <1>
1637                              <1> ;--------------------------------------------------------------------------
1638                              <1> ; SEND_RATE
1639                              <1> ;     SENDS DATA RATE COMMAND TO NEC
1640                              <1> ; ON ENTRY:  DI = DRIVE #
1641                              <1> ; ON EXIT:   NONE
1642                              <1> ; REGISTERS ALTERED: DX
1643                              <1> ;--------------------------------------------------------------------------
1644                              <1> SEND_RATE:
1645 00004B30 6650                <1>       PUSH  AX                  ; SAVE REG.
1646 00004B32 8025[A88A0100]3F    <1>       AND   byte [LASTRATE], ~SEND_MSK ; ELSE CLEAR LAST RATE ATTEMPTED
1647 00004B39 8A87[AD8A0100]      <1>       MOV   AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
1648 00004B3F 24C0                <1>       AND   AL,SEND_MSK         ; KEEP ONLY RATE BITS
1649 00004B41 0805[A88A0100]      <1>       OR    [LASTRATE], AL            ; SAVE NEW RATE FOR NEXT CHECK
1650 00004B47 C0C002              <1>       ROL   AL,2                ; MOVE TO BIT OUTPUT POSITIONS
1651 00004B4A 66BAF703            <1>       MOV   DX,03F7H            ; OUTPUT NEW DATA RATE
1652 00004B4E EE                  <1>       OUT   DX,AL
1653 00004B4F 6658                <1>       POP   AX                  ; RESTORE REG.
1654 00004B51 C3                  <1>       RETn
1655                              <1>
1656                              <1> ;--------------------------------------------------------------------------
1657                              <1> ; CHK_LASTRATE
1658                              <1> ;     CHECK PREVIOUS DATE RATE SNT TO THE CONTROLLER.
1659                              <1> ; ON ENTRY:
1660                              <1> ;     DI = DRIVE #
```

```
                        <1> ; ON EXIT:
                        <1> ;     ZF =  1 DATA RATE IS THE SAME AS THE LAST RATE SENT TO NEC
                        <1> ;     ZF =  0 DATA RATE IS DIFFERENT FROM LAST RATE
                        <1> ; REGISTERS ALTERED: DX
                        <1> ;--------------------------------------------------------------------------------
                        <1> CHK_LASTRATE:
00004B52 6650           <1>       PUSH   AX                  ; SAVE REG
00004B54 2225[A88A0100] <1>       AND    AH, [LASTRATE]            ; GET LAST DATA RATE SELECTED
00004B5A 8A87[AD8A0100] <1>       MOV    AL, [DSK_STATE+eDI] ; GET RATE STATE OF THIS DRIVE
00004B60 6625C0C0       <1>         AND     AX, SEND_MSK*257        ; KEEP ONLY RATE BITS OF BOTH
00004B64 38E0           <1>       CMP    AL, AH              ; COMPARE TO PREVIOUSLY TRIED
                        <1>                                  ; ZF = 1 RATE IS THE SAME
00004B66 6658           <1>       POP    AX                  ; RESTORE REG.
00004B68 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
00004B69 8B5504         <1>       mov    edx, [ebp+4]        ; Buffer address
00004B6C F7C2000000FF   <1>       test   edx, 0FF000000h           ; 16 MB limit (22/08/2015, bugfix)
00004B72 756E           <1>       jnz    short dma_bnd_err_stc
                        <1>       ;
00004B74 6650           <1>       push   ax                  ; DMA command
00004B76 52             <1>       push   edx                 ; *
00004B77 B203           <1>       mov    dl, 3               ; GET BYTES/SECTOR PARAMETER
00004B79 E851030000     <1>       call   GET_PARM            ;
00004B7E 88E1           <1>       mov    cl, ah                   ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
00004B80 6689F0         <1>       mov    ax, si              ; Sector count
00004B83 88C4           <1>       mov    ah, al              ; AH =  # OF SECTORS
00004B85 28C0           <1>       sub    al, al              ; AL = 0, AX = # SECTORS * 256
00004B87 66D1E8         <1>       shr    ax, 1               ; AX = # SECTORS * 128
00004B8A 66D3E0         <1>       shl    ax, cl              ; SHIFT BY PARAMETER VALUE
00004B8D 6648           <1>       dec    ax                  ; -1 FOR DMA VALUE
00004B8F 6689C1         <1>       mov    cx, ax
00004B92 5A             <1>       pop    edx                 ; *
00004B93 6658           <1>       pop    ax
00004B95 3C42           <1>       cmp    al, 42h
00004B97 7507           <1>         jne     short NOT_VERF
00004B99 BA0000FF00     <1>       mov    edx, 0FF0000h
00004B9E EB08           <1>       jmp    short J33
                        <1> NOT_VERF:
00004BA0 6601CA         <1>       add    dx, cx              ; check for overflow
00004BA3 723E           <1>       jc     short dma_bnd_err
                        <1>       ;
00004BA5 6629CA         <1>       sub    dx, cx              ; Restore start address
                        <1> J33:
00004BA8 FA             <1>       CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
00004BA9 E60C           <1>       OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
                        <1>       IODELAY                          ; WAIT FOR I/O
00004BAB EB00           <2>  jmp short $+2
00004BAD EB00           <2>  jmp short $+2
00004BAF E60B           <1>       OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
00004BB1 89D0           <1>       mov    eax, edx            ; Buffer address
00004BB3 E604           <1>       OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
                        <1>       IODELAY                          ; WAIT FOR I/O
00004BB5 EB00           <2>  jmp short $+2
00004BB7 EB00           <2>  jmp short $+2
00004BB9 88E0           <1>       MOV    AL,AH
00004BBB E604           <1>       OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
00004BBD C1E810         <1>       shr    eax, 16
                        <1>       IODELAY                          ; I/O WAIT STATE
00004BC0 EB00           <2>  jmp short $+2
00004BC2 EB00           <2>  jmp short $+2
00004BC4 E681           <1>       OUT    081H,AL                  ; OUTPUT highest BITS TO PAGE REGISTER
                        <1>       IODELAY
00004BC6 EB00           <2>  jmp short $+2
00004BC8 EB00           <2>  jmp short $+2
00004BCA 6689C8         <1>       mov    ax, cx              ; Byte count - 1
00004BCD E605           <1>       OUT    DMA+5,AL            ; LOW BYTE OF COUNT
                        <1>       IODELAY                          ; WAIT FOR I/O
00004BCF EB00           <2>  jmp short $+2
00004BD1 EB00           <2>  jmp short $+2
00004BD3 88E0           <1>       MOV    AL, AH
00004BD5 E605           <1>       OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
                        <1>       IODELAY
00004BD7 EB00           <2>  jmp short $+2
00004BD9 EB00           <2>  jmp short $+2
00004BDB FB             <1>       STI                        ; RE-ENABLE INTERRUPTS
00004BDC B002           <1>       MOV    AL, 2               ; MODE FOR 8237
00004BDE E60A           <1>       OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
                        <1>
```

```
1754 00004BE0 F8                  <1>       clc    ; 04/02/2016
1755 00004BE1 C3                  <1>       retn
1756                              <1>
1757                              <1> dma_bnd_err_stc:
1758 00004BE2 F9                  <1>       stc
1759                              <1> dma_bnd_err:
1760 00004BE3 C605[A08A0100]09    <1>       MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
1761 00004BEA C3                  <1>       RETn                        ; CY SET BY ABOVE IF ERROR
1762                              <1>
1763                              <1> ;; 16/12/2014
1764                              <1> ;;    CLI                         ; DISABLE INTERRUPTS DURING DMA SET-UP
1765                              <1> ;;    OUT    DMA+12,AL            ; SET THE FIRST/LA5T F/F
1766                              <1> ;;    ;JMP   $+2                  ; WAIT FOR I/O
1767                              <1> ;;    IODELAY
1768                              <1> ;;    OUT    DMA+11,AL            ; OUTPUT THE MODE BYTE
1769                              <1> ;;    ;SIODELAY
1770                              <1> ;;      ;CMP AL, 42H                    ; DMA VERIFY COMMAND
1771                              <1> ;;      ;JNE short NOT_VERF             ; NO
1772                              <1> ;;      ;XOR AX, AX             ; START ADDRESS
1773                              <1> ;;      ;JMP SHORT J33
1774                              <1> ;;;NOT_VERF:
1775                              <1> ;;    ;MOV   AX,ES                ; GET THE ES VALUE
1776                              <1> ;;    ;ROL   AX,4                 ; ROTATE LEFT
1777                              <1> ;;    ;MOV   CH,AL                ; GET HIGHEST NIBBLE OF ES TO CH
1778                              <1> ;;    ;AND   AL,11110000B         ; ZERO THE LOW NIBBLE FROM SEGMENT
1779                              <1> ;;    ;ADD   AX,[BP+2]            ; TEST FOR CARRY FROM ADDITION
1780                              <1> ;;    mov    eax, [ebp+4] ; 06/02/2015
1781                              <1> ;;    ;JNC   short J33
1782                              <1> ;;    ;INC   CH                   ; CARRY MEANS HIGH 4 BITS MUST BE INC
1783                              <1> ;;;J33:
1784                              <1> ;;    PUSH   eAX                  ; SAVE START ADDRESS
1785                              <1> ;;    OUT    DMA+4,AL             ; OUTPUT LOW ADDRESS
1786                              <1> ;;    ;JMP   $+2                  ; WAIT FOR I/O
1787                              <1> ;;    IODELAY
1788                              <1> ;;    MOV    AL,AH
1789                              <1> ;;    OUT    DMA+4,AL             ; OUTPUT HIGH ADDRESS
1790                              <1> ;;    shr    eax, 16             ; 07/02/2015
1791                              <1> ;;    ;MOV   AL,CH                ; GET HIGH 4 BITS
1792                              <1> ;;    ;JMP   $+2                  ; I/O WAIT STATE
1793                              <1> ;;    IODELAY
1794                              <1> ;;    ;AND   AL,00001111B
1795                              <1> ;;    OUT    081H,AL                   ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
1796                              <1> ;;    ;SIODELAY
1797                              <1> ;;
1798                              <1> ;;;----- DETERMINE COUNT
1799                              <1> ;;    sub    eax, eax ; 08/02/2015
1800                              <1> ;;    MOV    AX, SI               ; AL =  # OF SECTORS
1801                              <1> ;;    XCHG   AL, AH               ; AH =  # OF SECTORS
1802                              <1> ;;    SUB    AL, AL               ; AL = 0, AX = # SECTORS * 256
1803                              <1> ;;    SHR    AX, 1                ; AX = # SECTORS * 128
1804                              <1> ;;    PUSH   AX                   ; SAVE # OF SECTORS * 128
1805                              <1> ;;    MOV    DL, 3                ; GET BYTES/SECTOR PARAMETER
1806                              <1> ;;    CALL   GET_PARM             ; "
1807                              <1> ;;    MOV    CL,AH                ; SHIFT COUNT (0=128, 1=256, 2=512 ETC)
1808                              <1> ;;    POP    AX                   ; AX = # SECTORS * 128
1809                              <1> ;;    SHL    AX,CL                ; SHIFT BY PARAMETER VALUE
1810                              <1> ;;    DEC    AX                   ; -1 FOR DMA VALUE
1811                              <1> ;;    PUSH   eAX  ; 08/02/2015    ; SAVE COUNT VALUE
1812                              <1> ;;    OUT    DMA+5,AL             ; LOW BYTE OF COUNT
1813                              <1> ;;    ;JMP   $+2                  ; WAIT FOR I/O
1814                              <1> ;;    IODELAY
1815                              <1> ;;    MOV    AL, AH
1816                              <1> ;;    OUT    DMA+5,AL             ; HIGH BYTE OF COUNT
1817                              <1> ;;    ;IODELAY
1818                              <1> ;;    STI                         ; RE-ENABLE INTERRUPTS
1819                              <1> ;;    POP    eCX  ; 08/02/2015    ; RECOVER COUNT VALUE
1820                              <1> ;;    POP    eAX  ; 08/02/2015    ; RECOVER ADDRESS VALUE
1821                              <1> ;;    ;ADD   AX, CX               ; ADD, TEST FOR 64K OVERFLOW
1822                              <1> ;;    add    ecx, eax ; 08/02/2015
1823                              <1> ;;    MOV    AL, 2                ; MODE FOR 8237
1824                              <1> ;;    ;JMP   $+2                  ; WAIT FOR I/O
1825                              <1> ;;    SIODELAY
1826                              <1> ;;    OUT    DMA+10, AL           ; INITIALIZE THE DISKETTE CHANNEL
1827                              <1> ;;    ;JNC   short NO_BAD         ; CHECK FOR ERROR
1828                              <1> ;;    jc     short dma_bnd_err ; 08/02/2015
1829                              <1> ;;    and    ecx, 0FFF00000h ; 16 MB limit
1830                              <1> ;;    jz     short NO_BAD
1831                              <1> ;;dma_bnd_err:
1832                              <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
1833                              <1> ;;NO_BAD:
1834                              <1> ;;    RETn                        ; CY SET BY ABOVE IF ERROR
1835                              <1>
1836                              <1> ;--------------------------------------------------------------------------------
1837                              <1> ; FMTDMA_SET
1838                              <1> ;     THIS ROUTINE SETS UP THE DMA CONTROLLER FOR A FORMAT OPERATION.
1839                              <1> ;
1840                              <1> ; ON ENTRY:  NOTHING REQUIRED
1841                              <1> ;
1842                              <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
1843                              <1> ;--------------------------------------------------------------------------------
1844                              <1>
1845                              <1> FMTDMA_SET:
1846                              <1> ;; 20/02/2015 modification
1847 00004BEB 8B5504             <1>       mov    edx, [ebp+4]        ; Buffer address
1848 00004BEE F7C20000F0FF       <1>       test   edx, 0FFF00000h           ; 16 MB limit
1849 00004BF4 75EC               <1>       jnz    short dma_bnd_err_stc
1850                             <1>       ;
1851 00004BF6 6652               <1>       push   dx                  ; *
1852 00004BF8 B204               <1>       mov    DL, 4               ; SECTORS/TRACK VALUE IN PARM TABLE
1853 00004BFA E8D0020000         <1>       call   GET_PARM            ; "
1854 00004BFF 88E0               <1>       mov    al, ah              ; AL = SECTORS/TRACK VALUE
1855 00004C01 28E4               <1>       sub    ah, ah              ; AX = SECTORS/TRACK VALUE
1856 00004C03 66C1E002           <1>       shl    ax, 2               ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
1857 00004C07 6648               <1>       dec    ax                  ; -1 FOR DMA VALUE
1858 00004C09 6689C1             <1>       mov    cx, ax
```

```
1859 00004C0C 665A              <1>       pop    dx                  ; *
1860 00004C0E 6601CA            <1>       add    dx, cx              ; check for overflow
1861 00004C11 72D0              <1>       jc     short dma_bnd_err
1862                            <1>       ;
1863 00004C13 6629CA            <1>       sub    dx, cx              ; Restore start address
1864                            <1>       ;
1865 00004C16 B04A              <1>       MOV    AL, 04AH            ; WILL WRITE TO THE DISKETTE
1866 00004C18 FA                <1>       CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
1867 00004C19 E60C              <1>       OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
1868                            <1>       IODELAY                          ; WAIT FOR I/O
1868 00004C1B EB00              <2>  jmp short $+2
1868 00004C1D EB00              <2>  jmp short $+2
1869 00004C1F E60B              <1>       OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
1870 00004C21 89D0              <1>       mov    eax, edx            ; Buffer address
1871 00004C23 E604              <1>       OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
1872                            <1>       IODELAY                          ; WAIT FOR I/O
1872 00004C25 EB00              <2>  jmp short $+2
1872 00004C27 EB00              <2>  jmp short $+2
1873 00004C29 88E0              <1>       MOV    AL,AH
1874 00004C2B E604              <1>       OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
1875 00004C2D C1E810            <1>       shr    eax, 16
1876                            <1>       IODELAY                          ; I/O WAIT STATE
1876 00004C30 EB00              <2>  jmp short $+2
1876 00004C32 EB00              <2>  jmp short $+2
1877 00004C34 E681              <1>       OUT    081H,AL                   ; OUTPUT highest BITS TO PAGE REGISTER
1878                            <1>       IODELAY
1878 00004C36 EB00              <2>  jmp short $+2
1878 00004C38 EB00              <2>  jmp short $+2
1879 00004C3A 6689C8            <1>       mov    ax, cx              ; Byte count - 1
1880 00004C3D E605              <1>       OUT    DMA+5,AL            ; LOW BYTE OF COUNT
1881                            <1>       IODELAY                          ; WAIT FOR I/O
1881 00004C3F EB00              <2>  jmp short $+2
1881 00004C41 EB00              <2>  jmp short $+2
1882 00004C43 88E0              <1>       MOV    AL, AH
1883 00004C45 E605              <1>       OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
1884                            <1>       IODELAY
1884 00004C47 EB00              <2>  jmp short $+2
1884 00004C49 EB00              <2>  jmp short $+2
1885 00004C4B FB                <1>       STI                        ; RE-ENABLE INTERRUPTS
1886 00004C4C B002              <1>       MOV    AL, 2               ; MODE FOR 8237
1887 00004C4E E60A              <1>       OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
1888 00004C50 C3                <1>       retn
1889                            <1>
1890                            <1> ;; 08/02/2015 - Protected Mode Modification
1891                            <1> ;;    MOV    AL, 04AH            ; WILL WRITE TO THE DISKETTE
1892                            <1> ;;    CLI                        ; DISABLE INTERRUPTS DURING DMA SET-UP
1893                            <1> ;;    OUT    DMA+12,AL           ; SET THE FIRST/LA5T F/F
1894                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
1895                            <1> ;;    IODELAY
1896                            <1> ;;    OUT    DMA+11,AL           ; OUTPUT THE MODE BYTE
1897                            <1> ;;    ;MOV   AX,ES               ; GET THE ES VALUE
1898                            <1> ;;    ;ROL   AX,4                ; ROTATE LEFT
1899                            <1> ;;    ;MOV   CH,AL               ; GET HIGHEST NIBBLE OF ES TO CH
1900                            <1> ;;    ;AND   AL,11110000B        ; ZERO THE LOW NIBBLE FROM SEGMENT
1901                            <1> ;;    ;ADD   AX,[BP+2]           ; TEST FOR CARRY FROM ADDITION
1902                            <1> ;;    ;JNC   short J33A
1903                            <1> ;;    ;INC   CH                  ; CARRY MEANS HIGH 4 BITS MUST BE INC
1904                            <1> ;;    mov    eax, [ebp+4] ; 08/02/2015
1905                            <1> ;;;J33A:
1906                            <1> ;;    PUSH   eAX ; 08/02/2015    ; SAVE START ADDRESS
1907                            <1> ;;    OUT    DMA+4,AL            ; OUTPUT LOW ADDRESS
1908                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
1909                            <1> ;;    IODELAY
1910                            <1> ;;    MOV    AL,AH
1911                            <1> ;;    OUT    DMA+4,AL            ; OUTPUT HIGH ADDRESS
1912                            <1> ;;    shr    eax, 16 ; 08/02/2015
1913                            <1> ;;    ;MOV   AL,CH               ; GET HIGH 4 BITS
1914                            <1> ;;    ;JMP   $+2                 ; I/O WAIT STATE
1915                            <1> ;;    IODELAY
1916                            <1> ;;    ;AND   AL,00001111B
1917                            <1> ;;    OUT    081H,AL                  ; OUTPUT HIGH 4 BITS TO PAGE REGISTER
1918                            <1> ;;
1919                            <1> ;;;----- DETERMINE COUNT
1920                            <1> ;;    sub    eax, eax ; 08/02/2015
1921                            <1> ;;    MOV    DL, 4               ; SECTORS/TRACK VALUE IN PARM TABLE
1922                            <1> ;;    CALL   GET_PARM            ; "
1923                            <1> ;;    XCHG   AL, AH              ; AL = SECTORS/TRACK VALUE
1924                            <1> ;;    SUB    AH, AH              ; AX = SECTORS/TRACK VALUE
1925                            <1> ;;    SHL    AX, 2               ; AX = SEC/TRK * 4 (OFFSET C,H,R,N)
1926                            <1> ;;    DEC    AX                  ; -1 FOR DMA VALUE
1927                            <1> ;;    PUSH   eAX    ; 08/02/2015 ; SAVE # OF BYTES TO BE TRANSFERED
1928                            <1> ;;    OUT    DMA+5,AL            ; LOW BYTE OF COUNT
1929                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
1930                            <1> ;;    IODELAY
1931                            <1> ;;    MOV    AL, AH
1932                            <1> ;;    OUT    DMA+5,AL            ; HIGH BYTE OF COUNT
1933                            <1> ;;    STI                        ; RE-ENABLE INTERRUPTS
1934                            <1> ;;    POP    eCX    ; 08/02/2015 ; RECOVER COUNT VALUE
1935                            <1> ;;    POP    eAX    ; 08/02/2015 ; RECOVER ADDRESS VALUE
1936                            <1> ;;    ;ADD   AX, CX              ; ADD, TEST FOR 64K OVERFLOW
1937                            <1> ;;    add    ecx, eax ; 08/02/2015
1938                            <1> ;;    MOV    AL, 2               ; MODE FOR 8237
1939                            <1> ;;    ;JMP   $+2                 ; WAIT FOR I/O
1940                            <1> ;;    SIODELAY
1941                            <1> ;;    OUT    DMA+10, AL          ; INITIALIZE THE DISKETTE CHANNEL
1942                            <1> ;;    ;JNC   short FMTDMA_OK           ; CHECK FOR ERROR
1943                            <1> ;;    jc     short fmtdma_bnd_err ; 08/02/2015
1944                            <1> ;;    and    ecx, 0FFF00000h  ; 16 MB limit
1945                            <1> ;;    jz     short FMTDMA_OK
1946                            <1> ;;    stc    ; 20/02/2015
1947                            <1> ;;fmtdma_bnd_err:
1948                            <1> ;;    MOV    byte [DSKETTE_STATUS], DMA_BOUNDARY ; SET ERROR
1949                            <1> ;;FMTDMA_OK:
1950                            <1> ;;    RETn                       ; CY SET BY ABOVE IF ERROR
1951                            <1>
```

```
                        <1> ;--------------------------------------------------------------------------------
                        <1> ; NEC_INIT
                        <1> ;     THIS ROUTINE SEEKS TO THE REQUESTED TRACK AND INITIALIZES
                        <1> ;     THE NEC FOR THE READ/WRITE/VERIFY/FORMAT OPERATION.
                        <1> ;
                        <1> ; ON ENTRY:  AH = NEC COMMAND TO BE PERFORMED
                        <1> ;
                        <1> ; ON EXIT:   @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
                        <1> ;--------------------------------------------------------------------------------
                        <1> NEC_INIT:
00004C51 6650           <1>       PUSH   AX                  ; SAVE NEC COMMAND
00004C53 E8BC020000     <1>       CALL   MOTOR_ON            ; TURN MOTOR ON FOR SPECIFIC DRIVE
                        <1>
                        <1> ;-----      DO THE SEEK OPERATION
                        <1>
00004C58 8A6D01         <1>       MOV    CH,[eBP+1]          ; CH = TRACK #
00004C5B E8AF030000     <1>       CALL   SEEK                ; MOVE TO CORRECT TRACK
00004C60 6658           <1>       POP    AX                  ; RECOVER COMMAND
00004C62 721E           <1>       JC     short ER_1          ; ERROR ON SEEK
00004C64 BB[824C0000]   <1>       MOV    eBX, ER_1           ; LOAD ERROR ADDRESS
00004C69 53             <1>       PUSH   eBX                 ; PUSH NEC_OUT ERROR RETURN
                        <1>
                        <1> ;-----      SEND OUT THE PARAMETERS TO THE CONTROLLER
                        <1>
00004C6A E866030000     <1>       CALL   NEC_OUTPUT          ; OUTPUT THE OPERATION COMMAND
00004C6F 6689F0         <1>       MOV    AX,SI               ; AH = HEAD #
00004C72 89FB           <1>       MOV    eBX,eDI                   ; BL = DRIVE #
00004C74 C0E402         <1>       SAL    AH,2                ; MOVE IT TO BIT 2
00004C77 80E404         <1>       AND    AH,00000100B        ; ISOLATE THAT BIT
00004C7A 08DC           <1>       OR     AH,BL               ; OR IN THE DRIVE NUMBER
00004C7C E854030000     <1>       CALL   NEC_OUTPUT          ; FALL THRU CY SET IF ERROR
00004C81 5B             <1>       POP    eBX                 ; THROW AWAY ERROR RETURN
                        <1> ER_1:
00004C82 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:
00004C83 B8[CE4C0000]   <1>       MOV    eAX, ER_2           ; LOAD ERROR ADDRESS
00004C88 50             <1>       PUSH   eAX                 ; PUSH NEC_OUT ERROR RETURN
00004C89 8A6501         <1>       MOV    AH,[eBP+1]          ; OUTPUT TRACK #
00004C8C E844030000     <1>       CALL   NEC_OUTPUT
00004C91 6689F0         <1>       MOV    AX,SI               ; OUTPUT HEAD #
00004C94 E83C030000     <1>       CALL   NEC_OUTPUT
00004C99 8A6500         <1>         MOV     AH,[eBP]                ; OUTPUT SECTOR #
00004C9C E834030000     <1>       CALL   NEC_OUTPUT
00004CA1 B203           <1>       MOV    DL,3                ; BYTES/SECTOR PARAMETER FROM BLOCK
00004CA3 E827020000     <1>       CALL   GET_PARM            ; ... TO THE NEC
00004CA8 E828030000     <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
00004CAD B204           <1>       MOV    DL,4                ; EOT PARAMETER FROM BLOCK
00004CAF E81B020000     <1>       CALL   GET_PARM            ; ... TO THE NEC
00004CB4 E81C030000     <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
00004CB9 8A6305         <1>         MOV     AH, [eBX+MD.GAP]        ; GET GAP LENGTH
                        <1> _R15:
00004CBC E814030000     <1>       CALL   NEC_OUTPUT
00004CC1 B206           <1>       MOV    DL,6                ; DTL PARAMETER PROM BLOCK
00004CC3 E807020000     <1>       CALL   GET_PARM            ;  TO THE NEC
00004CC8 E808030000     <1>       CALL   NEC_OUTPUT          ; OUTPUT TO CONTROLLER
00004CCD 58             <1>       POP    eAX                 ; THROW AWAY ERROR EXIT
                        <1> ER_2:
00004CCE 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>
00004CCF 56             <1>       PUSH   eSI                 ; SAVE HEAD #, # OF SECTORS
00004CD0 E80D040000     <1>       CALL   WAIT_INT            ; WAIT FOR THE INTERRUPT
00004CD5 9C             <1>       PUSHF
00004CD6 E837040000     <1>       CALL   RESULTS                   ; GET THE NEC STATUS
00004CDB 724B           <1>       JC     short SET_END_POP
00004CDD 9D             <1>       POPF
00004CDE 723E           <1>       JC     short SET_END       ; LOOK FOR ERROR
                        <1>
                        <1> ;-----      CHECK THE RESULTS RETURNED BY THE CONTROLLER
                        <1>
00004CE0 FC             <1>       CLD                        ; SET THE CORRECT DIRECTION
00004CE1 BE[A18A0100]   <1>       MOV    eSI, NEC_STATUS           ; POINT TO STATUS FIELD
00004CE6 AC             <1>       lodsb                      ; GET ST0
00004CE7 24C0           <1>       AND    AL,11000000B        ; TEST FOR NORMAL TERMINATION
00004CE9 7433           <1>       JZ     short SET_END
00004CEB 3C40           <1>       CMP    AL,01000000B        ; TEST FOR ABNORMAL TERMINATION
00004CED 7527           <1>       JNZ    short J18           ; NOT ABNORMAL, BAD NEC
                        <1>
                        <1> ;-----      ABNORMAL TERMINATION, FIND OUT WHY
                        <1>
00004CEF AC             <1>       lodsb                      ; GET ST1
00004CF0 D0E0           <1>       SAL    AL,1                ; TEST FOR EDT FOUND
00004CF2 B404           <1>       MOV    AH,RECORD_NOT_FND
00004CF4 7222           <1>       JC     short J19
00004CF6 C0E002         <1>       SAL    AL,2
00004CF9 B410           <1>       MOV    AH,BAD_CRC
```

```
2057 00004CFB 721B                <1>       JC      short J19
2058 00004CFD D0E0                <1>       SAL     AL,1                ; TEST FOR DMA OVERRUN
2059 00004CFF B408                <1>       MOV     AH,BAD_DMA
2060 00004D01 7215                <1>       JC      short J19
2061 00004D03 C0E002              <1>       SAL     AL,2                ; TEST FOR RECORD NOT FOUND
2062 00004D06 B404                <1>       MOV     AH,RECORD_NOT_FND
2063 00004D08 720E                <1>       JC      short J19
2064 00004D0A D0E0                <1>       SAL     AL,1
2065 00004D0C B403                <1>       MOV     AH,WRITE_PROTECT    ; TEST FOR WRITE_PROTECT
2066 00004D0E 7208                <1>       JC      short J19
2067 00004D10 D0E0                <1>       SAL     AL,1                ; TEST MISSING ADDRESS MARK
2068 00004D12 B402                <1>       MOV     AH,BAD_ADDR_MARK
2069 00004D14 7202                <1>       JC      short J19
2070                              <1>
2071                              <1> ;-----      NEC MUST HAVE FAILED
2072                              <1> J18:
2073 00004D16 B420                <1>       MOV     AH,BAD_NEC
2074                              <1> J19:
2075 00004D18 0825[A08A0100]      <1>       OR      [DSKETTE_STATUS], AH
2076                              <1> SET_END:
2077 00004D1E 803D[A08A0100]01    <1>       CMP     byte [DSKETTE_STATUS], 1 ; SET ERROR CONDITION
2078 00004D25 F5                  <1>       CMC
2079 00004D26 5E                  <1>       POP     eSI
2080 00004D27 C3                  <1>       RETn                        ; RESTORE HEAD #, # OF SECTORS
2081                              <1>
2082                              <1> SET_END_POP:
2083 00004D28 9D                  <1>       POPF
2084 00004D29 EBF3                <1>       JMP     SHORT SET_END
2085                              <1>
2086                              <1> ;--------------------------------------------------------------------------
2087                              <1> ; DSTATE:   ESTABLISH STATE UPON SUCCESSFUL OPERATION.
2088                              <1> ;--------------------------------------------------------------------------
2089                              <1> DSTATE:
2090 00004D2B 803D[A08A0100]00    <1>       CMP     byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
2091 00004D32 753E                <1>       JNZ     short SETBAC            ; IF ERROR JUMP
2092 00004D34 808F[AD8A0100]10    <1>       OR      byte [DSK_STATE+eDI],MED_DET ; NO ERROR, MARK MEDIA AS DETERMINED
2093 00004D3B F687[AD8A0100]04    <1>       TEST    byte [DSK_STATE+eDI],DRV_DET ; DRIVE DETERMINED ?
2094 00004D42 752E                <1>       JNZ     short SETBAC        ; IF DETERMINED NO TRY TO DETERMINE
2095 00004D44 8A87[AD8A0100]      <1>       MOV     AL,[DSK_STATE+eDI]  ; LOAD STATE
2096 00004D4A 24C0                <1>       AND     AL,RATE_MSK         ; KEEP ONLY RATE
2097 00004D4C 3C80                <1>       CMP     AL,RATE_250         ; RATE 250 ?
2098 00004D4E 751B                <1>       JNE     short M_12          ; NO, MUST BE 1.2M OR 1.44M DRIVE
2099                              <1>
2100                              <1> ;-----      CHECK IF IT IS 1.44M
2101                              <1>
2102 00004D50 E871010000          <1>       CALL    CMOS_TYPE           ; RETURN DRIVE TYPE IN (AL)
2103                              <1>       ;;20/02/2015
2104                              <1>       ;;JC    short M_12          ; CMOS BAD
2105 00004D55 7414                <1>       jz      short M_12 ;; 20/02/2015
2106 00004D57 3C04                <1>       CMP     AL, 4               ; 1.44MB DRIVE ?
2107 00004D59 7410                <1>       JE      short M_12          ; YES
2108                              <1> M_720:
2109 00004D5B 80A7[AD8A0100]FD    <1>       AND     byte [DSK_STATE+eDI], ~FMT_CAPA ; TURN OFF FORMAT CAPABILITY
2110 00004D62 808F[AD8A0100]04    <1>       OR      byte [DSK_STATE+eDI],DRV_DET  ; MARK DRIVE DETERMINED
2111 00004D69 EB07                <1>       JMP     SHORT SETBAC        ; BACK
2112                              <1> M_12:
2113 00004D6B 808F[AD8A0100]06    <1>       OR      byte [DSK_STATE+eDI],DRV_DET+FMT_CAPA
2114                              <1>                                   ; TURN ON DETERMINED & FMT CAPA
2115                              <1> SETBAC:
2116 00004D72 C3                  <1>       RETn
2117                              <1>
2118                              <1> ;--------------------------------------------------------------------------
2119                              <1> ; RETRY
2120                              <1> ;     DETERMINES WHETHER A RETRY IS NECESSARY.
2121                              <1> ;     IF RETRY IS REQUIRED THEN STATE INFORMATION IS UPDATED FOR RETRY.
2122                              <1> ;
2123                              <1> ; ON EXIT:  CY = 1 FOR RETRY, CY = 0 FOR NO RETRY
2124                              <1> ;--------------------------------------------------------------------------
2125                              <1> RETRY:
2126 00004D73 803D[A08A0100]00    <1>       CMP     byte [DSKETTE_STATUS],0   ; GET STATUS OF OPERATION
2127 00004D7A 7445                <1>       JZ      short NO_RETRY            ; SUCCESSFUL OPERATION
2128 00004D7C 803D[A08A0100]80    <1>       CMP     byte [DSKETTE_STATUS],TIME_OUT ; IF TIME OUT NO RETRY
2129 00004D83 743C                <1>       JZ      short NO_RETRY
2130 00004D85 8AA7[AD8A0100]      <1>       MOV     AH,[DSK_STATE+eDI]  ; GET MEDIA STATE OF DRIVE
2131 00004D8B F6C410              <1>       TEST    AH,MED_DET          ; ESTABLISHED/DETERMINED ?
2132 00004D8E 7531                <1>       JNZ     short NO_RETRY            ; IF ESTABLISHED STATE THEN TRUE ERROR
2133 00004D90 80E4C0              <1>       AND     AH,RATE_MSK         ; ISOLATE RATE
2134 00004D93 8A2D[A88A0100]      <1>       MOV     CH,[LASTRATE]       ; GET START OPERATION STATE
2135 00004D99 C0C504              <1>       ROL     CH,4                ; TO CORRESPONDING BITS
2136 00004D9C 80E5C0              <1>       AND     CH,RATE_MSK         ; ISOLATE RATE BITS
2137 00004D9F 38E5                <1>       CMP     CH,AH               ; ALL RATES TRIED
2138 00004DA1 741E                <1>       JE      short NO_RETRY            ; IF YES, THEN TRUE ERROR
2139                              <1>
2140                              <1> ;     SETUP STATE INDICATOR FOR RETRY ATTEMPT TO NEXT RATE
2141                              <1> ;      00000000B (500) -> 10000000B    (250)
2142                              <1> ;      10000000B (250) -> 01000000B    (300)
2143                              <1> ;      01000000B (300) -> 00000000B    (500)
2144                              <1>
2145 00004DA3 80FC01              <1>       CMP     AH,RATE_500+1       ; SET CY FOR RATE 500
2146 00004DA6 D0DC                <1>       RCR     AH,1                ; TO NEXT STATE
2147 00004DA8 80E4C0              <1>       AND     AH,RATE_MSK         ; KEEP ONLY RATE BITS
2148 00004DAB 80A7[AD8A0100]1F    <1>       AND     byte [DSK_STATE+eDI], ~(RATE_MSK+DBL_STEP)
2149                              <1>                                   ; RATE, DBL STEP OFF
2150 00004DB2 08A7[AD8A0100]      <1>       OR      [DSK_STATE+eDI],AH  ; TURN ON NEW RATE
2151 00004DB8 C605[A08A0100]00    <1>       MOV     byte [DSKETTE_STATUS],0   ; RESET STATUS FOR RETRY
2152 00004DBF F9                  <1>       STC                         ; SET CARRY FOR RETRY
2153 00004DC0 C3                  <1>       RETn                        ; RETRY RETURN
2154                              <1>
2155                              <1> NO_RETRY:
2156 00004DC1 F8                  <1>       CLC                         ; CLEAR CARRY NO RETRY
2157 00004DC2 C3                  <1>       RETn                        ; NO RETRY RETURN
2158                              <1>
2159                              <1> ;--------------------------------------------------------------------------
2160                              <1> ; NUM_TRANS
2161                              <1> ;     THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT WERE
```

```
                        <1> ;     ACTUALLY TRANSFERRED TO/FROM THE DISKETTE.
                        <1> ;
                        <1> ; ON ENTRY:  [BP+1] = TRACK
                        <1> ;            SI-HI  = HEAD
                        <1> ;            [BP]   = START SECTOR
                        <1> ;
                        <1> ; ON EXIT:   AL = NUMBER ACTUALLY TRANSFERRED
                        <1> ;--------------------------------------------------------------------------
                        <1> NUM_TRANS:
00004DC3 30C0           <1>       XOR    AL,AL               ; CLEAR FOR ERROR
00004DC5 803D[A08A0100]00 <1>     CMP    byte [DSKETTE_STATUS],0   ; CHECK FOR ERROR
00004DCC 752C           <1>       JNZ    NT_OUT              ; IF ERROR 0 TRANSFERRED
00004DCE B204           <1>       MOV    DL,4                ; SECTORS/TRACK OFFSET TO DL
00004DD0 E8FA000000     <1>       CALL   GET_PARM            ; AH = SECTORS/TRACK
00004DD5 8A1D[A68A0100] <1>       MOV    BL, [NEC_STATUS+5]  ; GET ENDING SECTOR
00004DDB 6689F1         <1>       MOV    CX,SI               ; CH = HEAD # STARTED
00004DDE 3A2D[A58A0100] <1>       CMP    CH, [NEC_STATUS+4]  ; GET HEAD ENDED UP ON
00004DE4 750D           <1>       JNZ    DIF_HD              ; IF ON SAME HEAD, THEN NO ADJUST
00004DE6 8A2D[A48A0100] <1>       MOV    CH, [NEC_STATUS+3]  ; GET TRACK ENDED UP ON
00004DEC 3A6D01         <1>       CMP    CH,[eBP+1]          ; IS IT ASKED FOR TRACK
00004DEF 7404           <1>       JZ     short SAME_TRK            ; IF SAME TRACK NO INCREASE
00004DF1 00E3           <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
                        <1> DIF_HD:
00004DF3 00E3           <1>       ADD    BL,AH               ; ADD SECTORS/TRACK
                        <1> SAME_TRK:
00004DF5 2A5D00         <1>       SUB    BL,[eBP]            ; SUBTRACT START FROM END
00004DF8 88D8           <1>       MOV    AL,BL               ; TO AL
                        <1> NT_OUT:
00004DFA 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:
00004DFB B202           <1>       MOV    DL,2                ; GET THE MOTOR WAIT PARAMETER
00004DFD 6650           <1>       PUSH   AX                  ; SAVE NUMBER TRANSFERRED
00004DFF E8CB000000     <1>       CALL   GET_PARM
00004E04 8825[9F8A0100] <1>       MOV    [MOTOR_COUNT],AH    ; STORE UPON RETURN
00004E0A 6658           <1>       POP    AX                  ; RESTORE NUMBER TRANSFERRED
00004E0C 8A25[A08A0100] <1>       MOV    AH, [DSKETTE_STATUS]      ; GET STATUS OF OPERATION
00004E12 08E4           <1>       OR     AH,AH               ; CHECK FOR ERROR
00004E14 7402           <1>       JZ     short NUN_ERR       ; NO ERROR
00004E16 30C0           <1>       XOR    AL,AL               ; CLEAR NUMBER RETURNED
                        <1> NUN_ERR:
00004E18 80FC01         <1>       CMP    AH,1                ; SET THE CARRY FLAG TO INDICATE
00004E1B F5             <1>       CMC                        ; SUCCESS OR FAILURE
00004E1C 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:
00004E1D 8AA7[AD8A0100] <1>       MOV    AH, [DSK_STATE+eDI] ; ACCESS STATE
00004E23 F6C410         <1>       TEST   AH,MED_DET          ; ESTABLISHED STATE ?
00004E26 757E           <1>       JNZ    short NO_DBL              ; IF ESTABLISHED THEN DOUBLE DONE
                        <1>
                        <1> ;-----      CHECK FOR TRACK 0 TO SPEED UP ACKNOWLEDGE OF UNFORMATTED DISKETTE
                        <1>
00004E28 C605[9D8A0100]00 <1>     MOV    byte [SEEK_STATUS],0      ; SET RECALIBRATE REQUIRED ON ALL DRIVES
00004E2F E8E0000000     <1>       CALL   MOTOR_ON            ; ENSURE MOTOR STAY ON
00004E34 B500           <1>       MOV    CH,0                ; LOAD TRACK 0
00004E36 E8D4010000     <1>       CALL   SEEK                ; SEEK TO TRACK 0
00004E3B E868000000     <1>       CALL   READ_ID                   ; READ ID FUNCTION
00004E40 7249           <1>       JC     short SD_ERR        ; IF ERROR NO TRACK 0
                        <1>
                        <1> ;-----      INITIALIZE START AND MAX TRACKS (TIMES 2 FOR BOTH HEADS)
                        <1>
00004E42 66B95004       <1>       MOV    CX,0450H            ; START, MAX TRACKS
00004E46 F687[AD8A0100]01 <1>     TEST   byte [DSK_STATE+eDI],TRK_CAPA ; TEST FOR 80 TRACK CAPABILITY
00004E4D 7402           <1>       JZ     short CNT_OK        ; IF NOT COUNT IS SETUP
00004E4F 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:
00004E51 C605[9F8A0100]FF <1>       MOV     byte [MOTOR_COUNT], 0FFH ; ENSURE MOTOR STAYS ON FOR OPERATION
00004E58 6651           <1>       PUSH   CX                  ; SAVE TRACK, COUNT
00004E5A C605[A08A0100]00 <1>     MOV    byte [DSKETTE_STATUS],0   ; CLEAR STATUS, EXPECT ERRORS
00004E61 6631C0         <1>       XOR    AX,AX               ; CLEAR AX
00004E64 D0ED           <1>       SHR    CH,1                ; HALVE TRACK, CY = HEAD
00004E66 C0D003         <1>       RCL    AL,3                ; AX = HEAD IN CORRECT BIT
00004E69 6650           <1>       PUSH   AX                  ; SAVE HEAD
00004E6B E89F010000     <1>       CALL   SEEK                ; SEEK TO TRACK
00004E70 6658           <1>       POP    AX                  ; RESTORE HEAD
00004E72 6609C7         <1>       OR     DI,AX               ; DI = HEAD OR'ED DRIVE
00004E75 E82E000000     <1>       CALL   READ_ID                   ; READ ID HEAD 0
00004E7A 9C             <1>       PUSHF                      ; SAVE RETURN FROM READ_ID
00004E7B 6681E7FB00     <1>       AND    DI,11111011B        ; TURN OFF HEAD 1 BIT
00004E80 9D             <1>       POPF                       ; RESTORE ERROR RETURN
00004E81 6659           <1>       POP    CX                  ; RESTORE COUNT
00004E83 7308           <1>       JNC    short DO_CHK        ; IF OK, ASKED = RETURNED TRACK ?
00004E85 FEC5           <1>       INC    CH                  ; INC FOR NEXT TRACK
00004E87 38CD           <1>       CMP    CH,CL               ; REACHED MAXIMUM YET
```

```
2267 00004E89 75C6                <1>       JNZ    short CNT_OK        ; CONTINUE TILL ALL TRIED
2268                              <1>
2269                              <1> ;-----      FALL THRU, READ ID FAILED FOR ALL TRACKS
2270                              <1>
2271                              <1> SD_ERR:
2272 00004E8B F9                  <1>       STC                        ; SET CARRY FOR ERROR
2273 00004E8C C3                  <1>       RETn                       ; SETUP_DBL ERROR EXIT
2274                              <1>
2275                              <1> DO_CHK:
2276 00004E8D 8A0D[A48A0100]      <1>       MOV    CL, [NEC_STATUS+3]  ; LOAD RETURNED TRACK
2277 00004E93 888F[B18A0100]      <1>       MOV    [DSK_TRK+eDI], CL   ; STORE TRACK NUMBER
2278 00004E99 D0ED                <1>       SHR    CH,1                ; HALVE TRACK
2279 00004E9B 38CD                <1>       CMP    CH,CL               ; IS IT THE SAME AS ASKED FOR TRACK
2280 00004E9D 7407                <1>       JZ     short NO_DBL        ; IF SAME THEN NO DOUBLE STEP
2281 00004E9F 808F[AD8A0100]20    <1>       OR     byte [DSK_STATE+eDI],DBL_STEP ; TURN ON DOUBLE STEP REQUIRED
2282                              <1> NO_DBL:
2283 00004EA6 F8                  <1>       CLC                        ; CLEAR ERROR FLAG
2284 00004EA7 C3                  <1>       RETn
2285                              <1>
2286                              <1> ;--------------------------------------------------------------------------------
2287                              <1> ; READ_ID
2288                              <1> ;     READ ID FUNCTION.
2289                              <1> ;
2290                              <1> ; ON ENTRY:  DI : BIT 2 = HEAD; BITS 1,0 = DRIVE
2291                              <1> ;
2292                              <1> ; ON EXIT:   DI : BIT 2 IS RESET, BITS 1,0 = DRIVE
2293                              <1> ;            @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION
2294                              <1> ;--------------------------------------------------------------------------------
2295                              <1> READ_ID:
2296 00004EA8 B8[C54E0000]        <1>       MOV    eAX, ER_3           ; MOVE NEC OUTPUT ERROR ADDRESS
2297 00004EAD 50                  <1>       PUSH   eAX
2298 00004EAE B44A                <1>       MOV    AH,4AH              ; READ ID COMMAND
2299 00004EB0 E820010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
2300 00004EB5 6689F8              <1>       MOV    AX,DI               ; DRIVE # TO AH, HEAD 0
2301 00004EB8 88C4                <1>       MOV    AH,AL
2302 00004EBA E816010000          <1>       CALL   NEC_OUTPUT          ; TO CONTROLLER
2303 00004EBF E80BFEFFFF          <1>       CALL   NEC_TERM            ; WAIT FOR OPERATION, GET STATUS
2304 00004EC4 58                  <1>       POP    eAX                 ; THROW AWAY ERROR ADDRESS
2305                              <1> ER_3:
2306 00004EC5 C3                  <1>       RETn
2307                              <1>
2308                              <1> ;--------------------------------------------------------------------------------
2309                              <1> ; CMOS_TYPE
2310                              <1> ;     RETURNS DISKETTE TYPE FROM CMOS
2311                              <1> ;
2312                              <1> ; ON ENTRY:  DI = DRIVE #
2313                              <1> ;
2314                              <1> ; ON EXIT:   AL = TYPE; CY REFLECTS STATUS
2315                              <1> ;--------------------------------------------------------------------------------
2316                              <1>
2317                              <1> CMOS_TYPE: ; 11/12/2014
2318 00004EC6 8A87[0E6E0000]      <1> mov   al, [eDI+fd0_type]
2319 00004ECC 20C0                <1> and   al, al ; 18/12/2014
2320 00004ECE C3                  <1> retn
2321                              <1>
2322                              <1> ;CMOS_TYPE:
2323                              <1> ;     MOV    AL, CMOS_DIAG       ; CMOS DIAGNOSTIC STATUS BYTE ADDRESS
2324                              <1> ;     CALL   CMOS_READ           ; GET CMOS STATUS
2325                              <1> ;     TEST   AL,BAD_BAT+BAD_CKSUM      ; BATTERY GOOD AND CHECKSUM VALID
2326                              <1> ;     STC                        ; SET CY = 1 INDICATING ERROR FOR RETURN
2327                              <1> ;     JNZ    short BAD_CM        ; ERROR IF EITHER BIT ON
2328                              <1> ;     MOV    AL,CMOS_DISKETTE    ; ADDRESS OF DISKETTE BYTE IN CMOS
2329                              <1> ;     CALL   CMOS_READ           ; GET DISKETTE BYTE
2330                              <1> ;     OR     DI,DI               ; SEE WHICH DRIVE IN QUESTION
2331                              <1> ;     JNZ    short TB            ; IF DRIVE 1, DATA IN LOW NIBBLE
2332                              <1> ;     ROR    AL,4                ; EXCHANGE NIBBLES IF SECOND DRIVE
2333                              <1> ;TB:
2334                              <1> ;     AND    AL,0FH              ; KEEP ONLY DRIVE DATA, RESET CY, 0
2335                              <1> ;BAD_CM:
2336                              <1> ;     RETn                       ; CY, STATUS OF READ
2337                              <1>
2338                              <1> ;--------------------------------------------------------------------------------
2339                              <1> ; GET_PARM
2340                              <1> ;     THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
2341                              <1> ;     BLOCK POINTED TO BY THE DATA VARIABLE @DISK_POINTER. A BYTE FROM
2342                              <1> ;     THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
2343                              <1> ;     THE PARAMETER IN DL.
2344                              <1> ;
2345                              <1> ; ON ENTRY:  DL = INDEX OF BYTE TO BE FETCHED
2346                              <1> ;
2347                              <1> ; ON EXIT:   AH = THAT BYTE FROM BLOCK
2348                              <1> ;            AL,DH DESTROYED
2349                              <1> ;--------------------------------------------------------------------------------
2350                              <1> GET_PARM:
2351                              <1>       ;PUSH  DS
2352 00004ECF 56                  <1>       PUSH   eSI
2353                              <1>       ;SUB   AX,AX               ; DS = 0, BIOS DATA AREA
2354                              <1>       ;MOV   DS,AX
2355                              <1>       ;;mov  ax, cs
2356                              <1>       ;;mov  ds, ax
2357                              <1>       ; 08/02/2015 (protected mode modifications, bx -> ebx)
2358 00004ED0 87D3                <1>       XCHG   eDX,eBX                   ; BL = INDEX
2359                              <1>       ;SUB   BH,BH               ; BX = INDEX
2360 00004ED2 81E3FF000000        <1>       and    ebx, 0FFh
2361                              <1>       ;LDS   SI, [DISK_POINTER]  ; POINT TO BLOCK
2362                              <1>       ;
2363                              <1>       ; 17/12/2014
2364 00004ED8 66A1[FD6D0000]      <1>       mov    ax, [cfd] ; current (AL) and previous fd (AH)
2365 00004EDE 38E0                <1>       cmp    al, ah
2366 00004EE0 7425                <1>       je     short gpndc
2367 00004EE2 A2[FE6D0000]        <1>       mov    [pfd], al ; current drive -> previous drive
2368 00004EE7 53                  <1>       push   ebx ; 08/02/2015
2369 00004EE8 88C3                <1>       mov    bl, al
2370                              <1>       ; 11/12/2014
2371 00004EEA 8A83[0E6E0000]      <1>       mov    al, [eBX+fd0_type]  ; Drive type (0,1,2,3,4)
```

```
                        <1>       ; 18/12/2014
00004EF0 20C0           <1>       and    al, al
00004EF2 7507           <1>       jnz    short gpdtc
00004EF4 BB[E76D0000]   <1>       mov    ebx, MD_TBL6        ; 1.44 MB param. tbl. (default)
00004EF9 EB05           <1>         jmp     short gpdpu
                        <1> gpdtc:
00004EFB E817F9FFFF     <1>       call   DR_TYPE_CHECK
                        <1>       ; cf = 1 -> eBX points to 1.44MB fd parameter table (default)
                        <1> gpdpu:
00004F00 891D[846D0000] <1>       mov    [DISK_POINTER], ebx
00004F06 5B             <1>       pop    ebx
                        <1> gpndc:
00004F07 8B35[846D0000] <1>       mov    esi, [DISK_POINTER] ; 08/02/2015, si -> esi
00004F0D 8A241E         <1>       MOV    AH, [eSI+eBX]       ; GET THE WORD
00004F10 87D3           <1>       XCHG   eDX,eBX                   ; RESTORE BX
00004F12 5E             <1>       POP    eSI
                        <1>       ;POP   DS
00004F13 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; MOTOR_ON
                        <1> ;     TURN MOTOR ON AND WAIT FOR MOTOR START UP TIME. THE @MOTOR_COUNT
                        <1> ;     IS REPLACED WITH A SUFFICIENTLY HIGH NUMBER (0FFH) TO ENSURE
                        <1> ;     THAT THE MOTOR DOES NOT GO OFF DURING THE OPERATION. IF THE
                        <1> ;     MOTOR NEEDED TO BE TURNED ON, THE MULTI-TASKING HOOK FUNCTION
                        <1> ;     (AX=90FDH, INT 15) IS CALLED TELLING THE OPERATING SYSTEM
                        <1> ;     THAT THE BIOS IS ABOUT TO WAIT FOR MOTOR START UP. IF THIS
                        <1> ;     FUNCTION RETURNS WITH CY = 1, IT MEANS THAT THE MINIMUM WAIT
                        <1> ;     HAS BEEN COMPLETED. AT THIS POINT A CHECK IS MADE TO ENSURE
                        <1> ;     THAT THE MOTOR WASN'T TURNED OFF BY THE TIMER. IF THE HOOK DID
                        <1> ;     NOT WAIT, THE WAIT FUNCTION (AH=086H) IS CALLED TO WAIT THE
                        <1> ;     PRESCRIBED AMOUNT OF TIME. IF THE CARRY FLAG IS SET ON RETURN,
                        <1> ;     IT MEANS THAT THE FUNCTION IS IN USE AND DID NOT PERFORM THE
                        <1> ;     WAIT. A TIMER 1 WAIT LOOP WILL THEN DO THE WAIT.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE #
                        <1> ; ON EXIT:   AX,CX,DX DESTROYED
                        <1> ;--------------------------------------------------------------------------
                        <1> MOTOR_ON:
00004F14 53             <1>       PUSH   eBX                 ; SAVE REG.
00004F15 E82A000000     <1>       CALL   TURN_ON                   ; TURN ON MOTOR
00004F1A 7226           <1>       JC     short MOT_IS_ON           ; IF CY=1 NO WAIT
00004F1C E89BF9FFFF     <1>       CALL   XLAT_OLD           ; TRANSLATE STATE TO COMPATIBLE MODE
00004F21 E865F9FFFF     <1>       CALL   XLAT_NEW           ; TRANSLATE STATE TO PRESENT ARCH,
                        <1>       ;CALL  TURN_ON            ; CHECK AGAIN IF MOTOR ON
                        <1>       ;JC    MOT_IS_ON          ; IF NO WAIT MEANS IT IS ON
                        <1> M_WAIT:
00004F26 B20A           <1>       MOV    DL,10              ; GET THE MOTOR WAIT PARAMETER
00004F28 E8A2FFFFFF     <1>       CALL   GET_PARM
                        <1>       ;MOV   AL,AH              ; AL = MOTOR WAIT PARAMETER
                        <1>       ;XOR   AH,AH              ; AX = MOTOR WAIT PARAMETER
                        <1>       ;CMP   AL,8               ; SEE IF AT LEAST A SECOND IS SPECIFIED
00004F2D 80FC08         <1>       cmp    ah, 8
                        <1>       ;JAE   short GP2          ; IF YES, CONTINUE
00004F30 7702           <1>       ja     short J13
                        <1>       ;MOV   AL,8               ; ONE SECOND WAIT FOR MOTOR START UP
00004F32 B408           <1>       mov    ah, 8
                        <1>
                        <1> ;-----      AS CONTAINS NUMBER OF 1/8 SECONDS (125000 MICROSECONDS) TO WAIT
                        <1> GP2:
                        <1> ;-----      FOLLOWING LOOPS REQUIRED WHEN RTC WAIT FUNCTION IS ALREADY IN USE
                        <1> J13:                           ; WAIT FOR 1/8 SECOND PER (AL)
00004F34 B95E200000     <1>       MOV    eCX,8286           ; COUNT FOR 1/8 SECOND AT 15.085737 US
00004F39 E810D5FFFF     <1>       CALL   WAITF              ; GO TO FIXED WAIT ROUTINE
                        <1>       ;DEC   AL                 ; DECREMENT TIME VALUE
00004F3E FECC           <1>       dec    ah
00004F40 75F2           <1>       JNZ    short J13          ; ARE WE DONE YET
                        <1> MOT_IS_ON:
00004F42 5B             <1>       POP    eBX                ; RESTORE REG.
00004F43 C3             <1>       RETn
                        <1>
                        <1> ;--------------------------------------------------------------------------
                        <1> ; TURN_ON
                        <1> ;     TURN MOTOR ON AND RETURN WAIT STATE.
                        <1> ;
                        <1> ; ON ENTRY:  DI = DRIVE #
                        <1> ;
                        <1> ; ON EXIT:   CY = 0 MEANS WAIT REQUIRED
                        <1> ;            CY = 1 MEANS NO WAIT REQUIRED
                        <1> ;            AX,BX,CX,DX DESTROYED
                        <1> ;--------------------------------------------------------------------------
                        <1> TURN_ON:
00004F44 89FB           <1>       MOV    eBX,eDI                   ; BX = DRIVE #
00004F46 88D9           <1>       MOV    CL,BL              ; CL = DRIVE #
00004F48 C0C304         <1>       ROL    BL,4               ; BL = DRIVE SELECT
00004F4B FA             <1>       CLI                       ; NO INTERRUPTS WHILE DETERMINING STATUS
00004F4C C605[9F8A0100]FF <1>     MOV    byte [MOTOR_COUNT],0FFH   ; ENSURE MOTOR STAYS ON FOR OPERATION
00004F53 A0[9E8A0100]   <1>       MOV    AL, [MOTOR_STATUS] ; GET DIGITAL OUTPUT REGISTER REFLECTION
00004F58 2430           <1>       AND    AL,00110000B       ; KEEP ONLY DRIVE SELECT BITS
00004F5A B401           <1>       MOV    AH,1               ; MASK FOR DETERMINING MOTOR BIT
00004F5C D2E4           <1>       SHL    AH,CL              ; AH = MOTOR ON, A=00000001, B=00000010
                        <1>
                        <1> ;  AL = DRIVE SELECT FROM @MOTOR_STATUS
                        <1> ;  BL = DRIVE SELECT DESIRED
                        <1> ;  AH = MOTOR ON MASK DESIRED
                        <1>
00004F5E 38D8           <1>       CMP    AL,BL              ; REQUESTED DRIVE ALREADY SELECTED ?
00004F60 7508           <1>       JNZ    short TURN_IT_ON   ; IF NOT SELECTED JUMP
00004F62 8425[9E8A0100] <1>       TEST   AH, [MOTOR_STATUS] ; TEST MOTOR ON BIT
00004F68 7535           <1>       JNZ    short NO_MOT_WAIT  ; JUMP IF MOTOR ON AND SELECTED
                        <1>
                        <1> TURN_IT_ON:
00004F6A 08DC           <1>       OR     AH,BL              ; AH = DRIVE SELECT AND MOTOR ON
00004F6C 8A3D[9E8A0100] <1>       MOV    BH,[MOTOR_STATUS]  ; SAVE COPY OF @MOTOR_STATUS BEFORE
00004F72 80E70F         <1>       AND    BH,00001111B       ; KEEP ONLY MOTOR BITS
```

```
2477 00004F75 8025[9E8A0100]CF    <1>       AND    byte [MOTOR_STATUS],11001111B ; CLEAR OUT DRIVE SELECT
2478 00004F7C 0825[9E8A0100]      <1>       OR     [MOTOR_STATUS],AH   ; OR IN DRIVE SELECTED AND MOTOR ON
2479 00004F82 A0[9E8A0100]        <1>       MOV    AL,[MOTOR_STATUS]   ; GET DIGITAL OUTPUT REGISTER REFLECTION
2480 00004F87 88C3                <1>       MOV    BL,AL               ; BL=@MOTOR_STATUS AFTER, BH=BEFORE
2481 00004F89 80E30F              <1>       AND    BL,00001111B        ; KEEP ONLY MOTOR BITS
2482 00004F8C FB                  <1>       STI                        ; ENABLE INTERRUPTS AGAIN
2483 00004F8D 243F                <1>       AND    AL,00111111B        ; STRIP AWAY UNWANTED BITS
2484 00004F8F C0C004              <1>       ROL    AL,4                ; PUT BITS IN DESIRED POSITIONS
2485 00004F92 0C0C                <1>       OR     AL,00001100B        ; NO RESET, ENABLE DMA/INTERRUPT
2486 00004F94 66BAF203            <1>       MOV    DX,03F2H            ; SELECT DRIVE AND TURN ON MOTOR
2487 00004F98 EE                  <1>       OUT    DX,AL
2488 00004F99 38FB                <1>       CMP    BL,BH               ; NEW MOTOR TURNED ON ?
2489                              <1>       ;JZ    short NO_MOT_WAIT   ; NO WAIT REQUIRED IF JUST SELECT
2490 00004F9B 7403                <1>       je     short no_mot_w1 ; 27/02/2015
2491 00004F9D F8                  <1>       CLC                        ; (re)SET CARRY MEANING WAIT
2492 00004F9E C3                  <1>       RETn
2493                              <1>
2494                              <1> NO_MOT_WAIT:
2495 00004F9F FB                  <1>       sti
2496                              <1> no_mot_w1: ; 27/02/2015
2497 00004FA0 F9                  <1>       STC                        ; SET NO WAIT REQUIRED
2498                              <1>       ;STI                       ; INTERRUPTS BACK ON
2499 00004FA1 C3                  <1>       RETn
2500                              <1>
2501                              <1> ;--------------------------------------------------------------------------
2502                              <1> ; HD_WAIT
2503                              <1> ;     WAIT FOR HEAD SETTLE TIME.
2504                              <1> ;
2505                              <1> ; ON ENTRY:  DI = DRIVE #
2506                              <1> ;
2507                              <1> ; ON EXIT:   AX,BX,CX,DX DESTROYED
2508                              <1> ;--------------------------------------------------------------------------
2509                              <1> HD_WAIT:
2510 00004FA2 B209                <1>       MOV    DL,9                ; GET HEAD SETTLE PARAMETER
2511 00004FA4 E826FFFFFF          <1>       CALL   GET_PARM
2512 00004FA9 08E4                <1>       or     ah, ah ; 17/12/2014
2513 00004FAB 7519                <1>       jnz    short DO_WAT
2514 00004FAD F605[9E8A0100]80    <1>         TEST    byte [MOTOR_STATUS],10000000B ; SEE IF A WRITE OPERATION
2515                              <1>       ;JZ    short ISNT_WRITE    ; IF NOT, DO NOT ENFORCE ANY VALUES
2516                              <1>       ;OR    AH,AH               ; CHECK FOR ANY WAIT?
2517                              <1>       ;JNZ   short DO_WAT        ; IF THERE DO NOT ENFORCE
2518 00004FB4 741E                <1>       jz     short HW_DONE
2519 00004FB6 B40F                <1>       MOV    AH,HD12_SETTLE            ; LOAD 1.2M HEAD SETTLE MINIMUM
2520 00004FB8 8A87[AD8A0100]      <1>       MOV    AL,[DSK_STATE+eDI]  ; LOAD STATE
2521 00004FBE 24C0                <1>       AND    AL,RATE_MSK         ; KEEP ONLY RATE
2522 00004FC0 3C80                <1>       CMP    AL,RATE_250         ; 1.2 M DRIVE ?
2523 00004FC2 7502                <1>       JNZ    short DO_WAT        ; DEFAULT HEAD SETTLE LOADED
2524                              <1> ;GP3:
2525 00004FC4 B414                <1>       MOV    AH,HD320_SETTLE           ; USE 320/360 HEAD SETTLE
2526                              <1> ;     JMP    SHORT DO_WAT
2527                              <1>
2528                              <1> ;ISNT_WRITE:
2529                              <1> ;     OR     AH,AH               ; CHECK FOR NO WAIT
2530                              <1> ;     JZ     short HW_DONE       ; IF NOT WRITE AND 0 ITS OK
2531                              <1>
2532                              <1> ;-----       AH CONTAINS NUMBER OF MILLISECONDS TO WAIT
2533                              <1> DO_WAT:
2534                              <1> ;     MOV    AL,AH               ; AL = # MILLISECONDS
2535                              <1> ;     ;XOR   AH,AH               ; AX = # MILLISECONDS
2536                              <1> J29:                             ;      1 MILLISECOND LOOP
2537                              <1>       ;mov   cx, WAIT_FDU_HEAD_SETTLE ; 33 ; 1 ms in 30 micro units.
2538 00004FC6 B942000000          <1>       MOV    eCX,66              ; COUNT AT 15.085737 US PER COUNT
2539 00004FCB E87ED4FFFF          <1>       CALL   WAITF               ; DELAY FOR 1 MILLISECOND
2540                              <1>       ;DEC   AL                  ; DECREMENT THE COUNT
2541 00004FD0 FECC                <1>       dec    ah
2542 00004FD2 75F2                <1>       JNZ    short J29           ; DO AL MILLISECOND # OF TIMES
2543                              <1> HW_DONE:
2544 00004FD4 C3                  <1>       RETn
2545                              <1>
2546                              <1> ;--------------------------------------------------------------------------
2547                              <1> ; NEC_OUTPUT
2548                              <1> ;     THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
2549                              <1> ;     FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
2550                              <1> ;     TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE AMOUNT
2551                              <1> ;     OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
2552                              <1> ;
2553                              <1> ; ON ENTRY:  AH = BYTE TO BE OUTPUT
2554                              <1> ;
2555                              <1> ; ON EXIT:   CY = 0  SUCCESS
2556                              <1> ;            CY = 1  FAILURE -- DISKETTE STATUS UPDATED
2557                              <1> ;                    IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
2558                              <1> ;                    HIGHER THAN THE CALLER OF NEC OUTPUT. THIS REMOVES THE
2559                              <1> ;                    REQUIREMENT OF TESTING AFTER EVERY CALL OF NEC_OUTPUT.
2560                              <1> ;            AX,CX,DX DESTROYED
2561                              <1> ;--------------------------------------------------------------------------
2562                              <1>
2563                              <1> ; 09/12/2014 [Erdogan Tan]
2564                              <1> ;     (from 'PS2 Hardware Interface Tech. Ref. May 88', Page 09-05.)
2565                              <1> ; Diskette Drive Controller Status Register (3F4h)
2566                              <1> ;     This read only register facilitates the transfer of data between
2567                              <1> ;     the system microprocessor and the controller.
2568                              <1> ; Bit 7 - When set to 1, the Data register is ready to transfer data
2569                              <1> ;       with the system micrprocessor.
2570                              <1> ; Bit 6 - The direction of data transfer. If this bit is set to 0,
2571                              <1> ;       the transfer is to the controller.
2572                              <1> ; Bit 5 - When this bit is set to 1, the controller is in the non-DMA mode.
2573                              <1> ; Bit 4 - When this bit is set to 1, a Read or Write command is being executed.
2574                              <1> ; Bit 3 - Reserved.
2575                              <1> ; Bit 2 - Reserved.
2576                              <1> ; Bit 1 - When this bit is set to 1, dskette drive 1 is in the seek mode.
2577                              <1> ; Bit 0 - When this bit is set to 1, dskette drive 1 is in the seek mode.
2578                              <1>
2579                              <1> ; Data Register (3F5h)
2580                              <1> ; This read/write register passes data, commands and parameters, and provides
2581                              <1> ; diskette status information.
```

```
                                <1>
                                <1> NEC_OUTPUT:
                                <1>       ;PUSH  BX                   ; SAVE REG.
00004FD5 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
00004FD9 B91B410000             <1>       mov    ecx, WAIT_FDU_SEND_LH   ; 16667 (27/02/2015)
                                <1> ;
                                <1> ;WFPS_OUTER_LP:
                                <1> ;     ;
                                <1> ;WFPS_CHECK_PORT:
                                <1> J23:
00004FDE EC                     <1>       IN     AL,DX                ; GET STATUS
00004FDF 24C0                   <1>       AND    AL,11000000B         ; KEEP STATUS AND DIRECTION
00004FE1 3C80                   <1>       CMP    AL,10000000B         ; STATUS 1 AND DIRECTION 0 ?
00004FE3 7418                   <1>       JZ     short J27            ; STATUS AND DIRECTION OK
                                <1> WFPS_HI:
00004FE5 E461                   <1>       IN     AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
00004FE7 A810                   <1>       TEST   AL,010H                   ; transition on memory
00004FE9 75FA                   <1>       JNZ    SHORT WFPS_HI        ; refresh.
                                <1> WFPS_LO:
00004FEB E461                   <1>       IN     AL, PORT_B           ; SYS1
00004FED A810                   <1>       TEST   AL,010H
00004FEF 74FA                   <1>       JZ     SHORT WFPS_LO
                                <1>       ;LOOP  SHORT WFPS_CHECK_PORT
00004FF1 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>
00004FF3 800D[A08A0100]80       <1>       OR     byte [DSKETTE_STATUS],TIME_OUT
                                <1>       ;POP   BX                   ; RESTORE REG.
00004FFA 58                     <1>       POP    eAX ; 08/02/2015     ; DISCARD THE RETURN ADDRESS
00004FFB F9                     <1>       STC                         ; INDICATE ERROR TO CALLER
00004FFC C3                     <1>       RETn
                                <1>
                                <1> ;-----      DIRECTION AND STATUS OK; OUTPUT BYTE
                                <1>
                                <1> J27:
00004FFD 88E0                   <1>       MOV    AL,AH                ; GET BYTE TO OUTPUT
00004FFF 6642                   <1>       INC    DX                   ; DATA PORT = STATUS PORT + 1
00005001 EE                     <1>       OUT    DX,AL                ; OUTPUT THE BYTE
                                <1>       ;;NEWIODELAY  ;; 27/02/2015
                                <1>       ; 27/02/2015
00005002 9C                     <1>       PUSHF                       ; SAVE FLAGS
00005003 B903000000             <1>       MOV    eCX, 3               ; 30 TO 45 MICROSECONDS WAIT FOR
00005008 E841D4FFFF             <1>       CALL   WAITF                ; NEC FLAGS UPDATE CYCLE
0000500D 9D                     <1>       POPF                        ; RESTORE FLAGS FOR EXIT
                                <1>       ;POP   BX                   ; RESTORE REG
0000500E 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:
0000500F 89FB                   <1>       MOV    eBX,eDI                  ; BX = DRIVE #
```

```
2687 00005011 B001              <1>       MOV   AL,1                ; ESTABLISH MASK FOR RECALIBRATE TEST
2688 00005013 86CB              <1>       XCHG  CL,BL               ; SET DRIVE VALULE INTO CL
2689 00005015 D2C0              <1>       ROL   AL,CL               ; SHIFT MASK BY THE DRIVE VALUE
2690 00005017 86CB              <1>       XCHG  CL,BL               ; RECOVER TRACK VALUE
2691 00005019 8405[9D8A0100]    <1>       TEST  AL,[SEEK_STATUS]    ; TEST FOR RECALIBRATE REQUIRED
2692 0000501F 7526              <1>       JNZ   short J28A          ; JUMP IF RECALIBRATE NOT REQUIRED
2693                            <1>
2694 00005021 0805[9D8A0100]    <1>       OR    [SEEK_STATUS],AL    ; TURN ON THE NO RECALIBRATE BIT IN FLAG
2695 00005027 E862000000        <1>       CALL  RECAL               ; RECALIBRATE DRIVE
2696 0000502C 730E              <1>       JNC   short AFT_RECAL           ; RECALIBRATE DONE
2697                            <1>
2698                            <1> ;-----      ISSUE RECALIBRATE FOR 80 TRACK DISKETTES
2699                            <1>
2700 0000502E C605[A08A0100]00  <1>       MOV   byte [DSKETTE_STATUS],0   ; CLEAR OUT INVALID STATUS
2701 00005035 E854000000        <1>       CALL  RECAL               ; RECALIBRATE DRIVE
2702 0000503A 7251              <1>       JC    short RB            ; IF RECALIBRATE FAILS TWICE THEN ERROR
2703                            <1>
2704                            <1> AFT_RECAL:
2705 0000503C C687[B18A0100]00  <1>         MOV    byte [DSK_TRK+eDI],0    ; SAVE NEW CYLINDER AS PRESENT POSITION
2706 00005043 08ED              <1>       OR    CH,CH               ; CHECK FOR SEEK TO TRACK 0
2707 00005045 743F              <1>       JZ    short DO_WAIT       ; HEAD SETTLE, CY = 0 IF JUMP
2708                            <1>
2709                            <1> ;-----      DRIVE IS IN SYNCHRONIZATION WITH CONTROLLER, SEEK TO TRACK
2710                            <1>
2711 00005047 F687[AD8A0100]20  <1> J28A: TEST  byte [DSK_STATE+eDI],DBL_STEP ; CHECK FOR DOUBLE STEP REQUIRED
2712 0000504E 7402              <1>       JZ    short _R7           ; SINGLE STEP REQUIRED BYPASS DOUBLE
2713 00005050 D0E5              <1>       SHL   CH,1                ; DOUBLE NUMBER OF STEP TO TAKE
2714                            <1>
2715 00005052 3AAF[B18A0100]    <1> _R7:  CMP   CH, [DSK_TRK+eDI]   ; SEE IF ALREADY AT THE DESIRED TRACK
2716 00005058 7433              <1>       JE    short RB            ; IF YES, DO NOT NEED TO SEEK
2717                            <1>
2718 0000505A BA[8D500000]      <1>       MOV   eDX, NEC_ERR        ; LOAD RETURN ADDRESS
2719 0000505F 52                <1>       PUSH  eDX ; (*)           ; ON STACK FOR NEC OUTPUT ERROR
2720 00005060 88AF[B18A0100]    <1>       MOV   [DSK_TRK+eDI],CH    ; SAVE NEW CYLINDER AS PRESENT POSITION
2721 00005066 B40F              <1>       MOV   AH,0FH              ; SEEK COMMAND TO NEC
2722 00005068 E868FFFFFF        <1>       CALL  NEC_OUTPUT
2723 0000506D 89FB              <1>       MOV   eBX,eDI                   ; BX = DRIVE #
2724 0000506F 88DC              <1>       MOV   AH,BL               ; OUTPUT DRIVE NUMBER
2725 00005071 E85FFFFFFF        <1>       CALL  NEC_OUTPUT
2726 00005076 8AA7[B18A0100]    <1>       MOV   AH, [DSK_TRK+eDI]   ; GET CYLINDER NUMBER
2727 0000507C E854FFFFFF        <1>       CALL  NEC_OUTPUT
2728 00005081 E829000000        <1>       CALL  CHK_STAT_2          ; ENDING INTERRUPT AND SENSE STATUS
2729                            <1>
2730                            <1> ;-----      WAIT FOR HEAD SETTLE
2731                            <1>
2732                            <1> DO_WAIT:
2733 00005086 9C                <1>       PUSHF                     ; SAVE STATUS
2734 00005087 E816FFFFFF        <1>       CALL  HD_WAIT                   ; WAIT FOR HEAD SETTLE TIME
2735 0000508C 9D                <1>       POPF                      ; RESTORE STATUS
2736                            <1> RB:
2737                            <1> NEC_ERR:
2738                            <1>       ; 08/02/2015 (code trick here from original IBM PC/AT DISKETTE.ASM)
2739                            <1>       ; (*) nec_err -> retn (push edx -> pop edx) -> nec_err -> retn
2740 0000508D C3                <1>       RETn                      ; RETURN TO CALLER
2741                            <1>
2742                            <1> ;--------------------------------------------------------------------------
2743                            <1> ; RECAL
2744                            <1> ;     RECALIBRATE DRIVE
2745                            <1> ;
2746                            <1> ; ON ENTRY:  DI = DRIVE #
2747                            <1> ;
2748                            <1> ; ON EXIT:   CY REFLECTS STATUS OF OPERATION.
2749                            <1> ;--------------------------------------------------------------------------
2750                            <1> RECAL:
2751 0000508E 6651              <1>       PUSH  CX
2752 00005090 B8[AC500000]      <1>       MOV   eAX, RC_BACK        ; LOAD NEC_OUTPUT ERROR
2753 00005095 50                <1>       PUSH  eAX
2754 00005096 B407              <1>       MOV   AH,07H              ; RECALIBRATE COMMAND
2755 00005098 E838FFFFFF        <1>       CALL  NEC_OUTPUT
2756 0000509D 89FB              <1>       MOV   eBX,eDI                   ; BX = DRIVE #
2757 0000509F 88DC              <1>       MOV   AH,BL
2758 000050A1 E82FFFFFFF        <1>       CALL  NEC_OUTPUT          ; OUTPUT THE DRIVE NUMBER
2759 000050A6 E804000000        <1>       CALL  CHK_STAT_2          ; GET THE INTERRUPT AND SENSE INT STATUS
2760 000050AB 58                <1>       POP   eAX                 ; THROW AWAY ERROR
2761                            <1> RC_BACK:
2762 000050AC 6659              <1>       POP   CX
2763 000050AE C3                <1>       RETn
2764                            <1>
2765                            <1> ;--------------------------------------------------------------------------
2766                            <1> ; CHK_STAT_2
2767                            <1> ;     THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER RECALIBRATE,
2768                            <1> ;     OR SEEK TO THE ADAPTER. THE INTERRUPT IS WAITED FOR, THE
2769                            <1> ;     INTERRUPT STATUS SENSED, AND THE RESULT RETURNED TO THE CALLER.
2770                            <1> ;
2771                            <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
2772                            <1> ;--------------------------------------------------------------------------
2773                            <1> CHK_STAT_2:
2774 000050AF B8[D7500000]      <1>         MOV    eAX, CS_BACK           ; LOAD NEC_OUTPUT ERROR ADDRESS
2775 000050B4 50                <1>       PUSH  eAX
2776 000050B5 E828000000        <1>       CALL  WAIT_INT            ; WAIT FOR THE INTERRUPT
2777 000050BA 721A              <1>       JC    short J34           ; IF ERROR, RETURN IT
2778 000050BC B408              <1>       MOV   AH,08H              ; SENSE INTERRUPT STATUS COMMAND
2779 000050BE E812FFFFFF        <1>       CALL  NEC_OUTPUT
2780 000050C3 E84A000000        <1>       CALL  RESULTS                   ; READ IN THE RESULTS
2781 000050C8 720C              <1>       JC    short J34
2782 000050CA A0[A18A0100]      <1>       MOV   AL,[NEC_STATUS]           ; GET THE FIRST STATUS BYTE
2783 000050CF 2460              <1>       AND   AL,01100000B        ; ISOLATE THE BITS
2784 000050D1 3C60              <1>       CMP   AL,01100000B        ; TEST FOR CORRECT VALUE
2785 000050D3 7403              <1>       JZ    short J35           ; IF ERROR, GO MARK IT
2786 000050D5 F8                <1>       CLC                       ; GOOD RETURN
2787                            <1> J34:
2788 000050D6 58                <1>       POP   eAX                 ; THROW AWAY ERROR RETURN
2789                            <1> CS_BACK:
2790 000050D7 C3                <1>       RETn
2791                            <1> J35:
```

```
2792 000050D8 800D[A08A0100]40    <1>       OR     byte [DSKETTE_STATUS], BAD_SEEK
2793 000050DF F9                  <1>       STC                          ; ERROR RETURN CODE
2794 000050E0 EBF4                <1>       JMP    SHORT J34
2795                              <1>
2796                              <1> ;--------------------------------------------------------------------------------
2797                              <1> ; WAIT_INT
2798                              <1> ;     THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR A TIME OUT ROUTINE
2799                              <1> ;     TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE RETURNED
2800                              <1> ;     IF THE DRIVE IS NOT READY.
2801                              <1> ;
2802                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
2803                              <1> ;--------------------------------------------------------------------------------
2804                              <1>
2805                              <1> ; 17/12/2014
2806                              <1> ; 2.5 seconds waiting !
2807                              <1> ;(AWARD BIOS - 1999, WAIT_FDU_INT_LOW, WAIT_FDU_INT_HI)
2808                              <1> ; amount of time to wait for completion interrupt from NEC.
2809                              <1>
2810                              <1>
2811                              <1> WAIT_INT:
2812 000050E2 FB                  <1>       STI                          ; TURN ON INTERRUPTS, JUST IN CASE
2813 000050E3 F8                  <1>       CLC                          ; CLEAR TIMEOUT INDICATOR
2814                              <1>        ;MOV  BL,10                 ; CLEAR THE COUNTERS
2815                              <1>        ;XOR  CX,CX                 ; FOR 2 SECOND WAIT
2816                              <1>
2817                              <1>       ; Modification from AWARD BIOS - 1999 (ATORGS.ASM, WAIT
2818                              <1>       ;
2819                              <1>       ;WAIT_FOR_MEM:
2820                              <1>       ;      Waits for a bit at a specified memory location pointed
2821                              <1>       ;      to by ES:[DI] to become set.
2822                              <1>       ;INPUT:
2823                              <1>       ;      AH=Mask to test with.
2824                              <1>       ;      ES:[DI] = memory location to watch.
2825                              <1>       ;      BH:CX=Number of memory refresh periods to delay.
2826                              <1>       ;           (normally 30 microseconds per period.)
2827                              <1>
2828                              <1>       ; waiting for (max.) 2.5 secs in 30 micro units.
2829                              <1> ;     mov    cx, WAIT_FDU_INT_LO       ; 017798
2830                              <1> ;;    mov    bl, WAIT_FDU_INT_HI
2831                              <1> ;     mov    bl, WAIT_FDU_INT_HI + 1
2832                              <1>       ; 27/02/2015
2833 000050E4 B986450100          <1>       mov    ecx, WAIT_FDU_INT_LH      ; 83334 (2.5 seconds)
2834                              <1> WFMS_CHECK_MEM:
2835 000050E9 F605[9D8A0100]80    <1>       test   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
2836 000050F0 7516                <1>         jnz     short J37
2837                              <1> WFMS_HI:
2838 000050F2 E461                <1>       IN     AL,PORT_B  ; 061h   ; SYS1, wait for lo to hi
2839 000050F4 A810                <1>       TEST   AL,010H                   ; transition on memory
2840 000050F6 75FA                <1>       JNZ    SHORT WFMS_HI       ; refresh.
2841                              <1> WFMS_LO:
2842 000050F8 E461                <1>       IN     AL,PORT_B           ;SYS1
2843 000050FA A810                <1>       TEST   AL,010H
2844 000050FC 74FA                <1>       JZ     SHORT WFMS_LO
2845 000050FE E2E9                <1>         LOOP    WFMS_CHECK_MEM
2846                              <1> ;WFMS_OUTER_LP:
2847                              <1> ;;    or     bl, bl              ; check outer counter
2848                              <1> ;;    jz     short J36A          ; WFMS_TIMEOUT
2849                              <1> ;     dec    bl
2850                              <1> ;     jz     short J36A
2851                              <1> ;     jmp    short WFMS_CHECK_MEM
2852                              <1>
2853                              <1>       ;17/12/2014
2854                              <1>       ;16/12/2014
2855                              <1> ;        mov     byte [wait_count], 0    ; Reset (INT 08H) counter
2856                              <1> ;J36:
2857                              <1> ;     TEST   byte [SEEK_STATUS],INT_FLAG ; TEST FOR INTERRUPT OCCURRING
2858                              <1> ;     JNZ    short J37
2859                              <1>       ;16/12/2014
2860                              <1>       ;LOOP  J36                 ; COUNT DOWN WHILE WAITING
2861                              <1>       ;DEC   BL                  ; SECOND LEVEL COUNTER
2862                              <1>       ;JNZ   short J36
2863                              <1> ;       cmp     byte [wait_count], 46   ; (46/18.2 seconds)
2864                              <1> ;     jb     short J36
2865                              <1>
2866                              <1> ;WFMS_TIMEOUT:
2867                              <1> ;J36A:
2868 00005100 800D[A08A0100]80    <1>       OR     byte [DSKETTE_STATUS], TIME_OUT ; NOTHING HAPPENED
2869 00005107 F9                  <1>       STC                          ; ERROR RETURN
2870                              <1> J37:
2871 00005108 9C                  <1>       PUSHF                        ; SAVE CURRENT CARRY
2872 00005109 8025[9D8A0100]7F    <1>       AND    byte [SEEK_STATUS], ~INT_FLAG ; TURN OFF INTERRUPT FLAG
2873 00005110 9D                  <1>       POPF                         ; RECOVER CARRY
2874 00005111 C3                  <1>       RETn                         ; GOOD RETURN CODE
2875                              <1>
2876                              <1> ;--------------------------------------------------------------------------------
2877                              <1> ; RESULTS
2878                              <1> ;     THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER RETURNS
2879                              <1> ;     FOLLOWING AN INTERRUPT.
2880                              <1> ;
2881                              <1> ; ON EXIT:  @DSKETTE_STATUS, CY REFLECT STATUS OF OPERATION.
2882                              <1> ;           AX,BX,CX,DX DESTROYED
2883                              <1> ;--------------------------------------------------------------------------------
2884                              <1> RESULTS:
2885 00005112 57                  <1>       PUSH   eDI
2886 00005113 BF[A18A0100]        <1>       MOV    eDI, NEC_STATUS           ; POINTER TO DATA AREA
2887 00005118 B307                <1>       MOV    BL,7                ; MAX STATUS BYTES
2888 0000511A 66BAF403            <1>       MOV    DX,03F4H            ; STATUS PORT
2889                              <1>
2890                              <1> ;-----      WAIT FOR REQUEST FOR MASTER
2891                              <1>
2892                              <1> _R10:
2893                              <1>       ; 16/12/2014
2894                              <1>       ; wait for (max) 0.5 seconds
2895                              <1>       ;MOV   BH,2                ; HIGH ORDER COUNTER
2896                              <1>       ;XOR   CX,CX               ; COUNTER
```

```
                                <1>
                                <1>       ;Time to wait while waiting for each byte of NEC results = .5
                                <1>       ;seconds.  .5 seconds = 500,000 micros.  500,000/30 = 16,667.
                                <1>       ; 27/02/2015
0000511E B91B410000             <1>       mov   ecx, WAIT_FDU_RESULTS_LH ; 16667
                                <1>       ;mov  cx, WAIT_FDU_RESULTS_LO  ; 16667
                                <1>       ;mov  bh, WAIT_FDU_RESULTS_HI+1 ; 0+1
                                <1>
                                <1> WFPSR_OUTER_LP:
                                <1>       ;
                                <1> WFPSR_CHECK_PORT:
                                <1> J39:                              ; WAIT FOR MASTER
00005123 EC                     <1>       IN    AL,DX                 ; GET STATUS
00005124 24C0                   <1>       AND   AL,11000000B          ; KEEP ONLY STATUS AND DIRECTION
00005126 3CC0                   <1>       CMP   AL,11000000B          ; STATUS 1 AND DIRECTION 1 ?
00005128 7418                   <1>       JZ    short J42             ; STATUS AND DIRECTION OK
                                <1> WFPSR_HI:
0000512A E461                   <1>       IN    AL, PORT_B   ;061h  ; SYS1 ; wait for hi to lo
0000512C A810                   <1>       TEST  AL,010H                   ; transition on memory
0000512E 75FA                   <1>       JNZ   SHORT WFPSR_HI            ; refresh.
                                <1> WFPSR_LO:
00005130 E461                   <1>       IN    AL, PORT_B            ; SYS1
00005132 A810                   <1>       TEST  AL,010H
00005134 74FA                   <1>       JZ    SHORT WFPSR_LO
00005136 E2EB                   <1>         LOOP    WFPSR_CHECK_PORT
                                <1>       ;; 27/02/2015
                                <1>       ;;dec  bh
                                <1>       ;;jnz  short WFPSR_OUTER_LP
                                <1>       ;jmp   short WFPSR_TIMEOUT ; fail
                                <1>
                                <1>       ;;mov  byte [wait_count], 0
                                <1> ;J39:                             ; WAIT FOR MASTER
                                <1> ;     IN    AL,DX                 ; GET STATUS
                                <1> ;     AND   AL,11000000B          ; KEEP ONLY STATUS AND DIRECTION
                                <1> ;     CMP   AL,11000000B          ; STATUS 1 AND DIRECTION 1 ?
                                <1> ;     JZ    short J42             ; STATUS AND DIRECTION OK
                                <1>       ;LOOP  J39                  ; LOOP TILL TIMEOUT
                                <1>       ;DEC   BH                   ; DECREMENT HIGH ORDER COUNTER
                                <1>       ;JNZ   short J39            ; REPEAT TILL DELAY DONE
                                <1>       ;
                                <1>       ;;cmp  byte [wait_count], 10  ; (10/18.2 seconds)
                                <1>       ;;jb   short J39
                                <1>
                                <1> ;WFPSR_TIMEOUT:
00005138 800D[A08A0100]80       <1>       OR    byte [DSKETTE_STATUS],TIME_OUT
0000513F F9                     <1>       STC                         ; SET ERROR RETURN
00005140 EB29                   <1>       JMP   SHORT POPRES          ; POP REGISTERS AND RETURN
                                <1>
                                <1> ;-----      READ IN THE STATUS
                                <1>
                                <1> J42:
00005142 EB00                   <1>       JMP   $+2                   ; I/O DELAY
00005144 6642                   <1>       INC   DX                    ; POINT AT DATA PORT
00005146 EC                     <1>       IN    AL,DX                 ; GET THE DATA
                                <1>       ; 16/12/2014
                                <1>       NEWIODELAY
00005147 E6EB                   <2>  out 0ebh,al
00005149 8807                   <1>         MOV     [eDI],AL                ; STORE THE BYTE
0000514B 47                     <1>       INC   eDI                   ; INCREMENT THE POINTER
                                <1>       ; 16/12/2014
                                <1> ;     push  cx
                                <1> ;     mov   cx, 30
                                <1> ;wdw2:
                                <1> ;     NEWIODELAY
                                <1> ;     loop  wdw2
                                <1> ;     pop   cx
                                <1>
0000514C B903000000             <1>       MOV   eCX,3                 ; MINIMUM 24 MICROSECONDS FOR NEC
00005151 E8F8D2FFFF             <1>       CALL  WAITF                 ; WAIT 30 TO 45 MICROSECONDS
00005156 664A                   <1>       DEC   DX                    ; POINT AT STATUS PORT
00005158 EC                     <1>       IN    AL,DX                 ; GET STATUS
                                <1>       ; 16/12/2014
                                <1>       NEWIODELAY
00005159 E6EB                   <2>  out 0ebh,al
                                <1>       ;
0000515B A810                   <1>       TEST  AL,00010000B          ; TEST FOR NEC STILL BUSY
0000515D 740C                   <1>       JZ    short POPRES          ; RESULTS DONE ?
                                <1>
0000515F FECB                   <1>       DEC   BL                    ; DECREMENT THE STATUS COUNTER
00005161 75BB                   <1>         JNZ     short _R10              ; GO BACK FOR MORE
00005163 800D[A08A0100]20       <1>       OR    byte [DSKETTE_STATUS],BAD_NEC ; TOO MANY STATUS BYTES
0000516A F9                     <1>       STC                         ; SET ERROR FLAG
                                <1>
                                <1> ;-----      RESULT OPERATION IS DONE
                                <1> POPRES:
0000516B 5F                     <1>       POP   eDI
0000516C C3                     <1>       RETn                        ; RETURN WITH CARRY SET
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; READ_DSKCHNG
                                <1> ;     READS THE STATE OF THE DISK CHANGE LINE.
                                <1> ;
                                <1> ; ON ENTRY:  DI = DRIVE #
                                <1> ;
                                <1> ; ON EXIT:   DI = DRIVE #
                                <1> ;            ZF = 0 : DISK CHANGE LINE INACTIVE
                                <1> ;            ZF = 1 : DISK CHANGE LINE ACTIVE
                                <1> ;            AX,CX,DX DESTROYED
                                <1> ;--------------------------------------------------------------------------------
                                <1> READ_DSKCHNG:
0000516D E8A2FDFFFF             <1>       CALL  MOTOR_ON              ; TURN ON THE MOTOR IF OFF
00005172 66BAF703               <1>       MOV   DX,03F7H              ; ADDRESS DIGITAL INPUT REGISTER
00005176 EC                     <1>       IN    AL,DX                 ; INPUT DIGITAL INPUT REGISTER
00005177 A880                   <1>       TEST  AL,DSK_CHG            ; CHECK FOR DISK CHANGE LINE ACTIVE
00005179 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:
0000517A E895FDFFFF             <1>       CALL   MOTOR_ON           ; TURN ON MOTOR IF NOT ALREADY ON
0000517F E80AFFFFFF             <1>       CALL   RECAL              ; RECALIBRATE DRIVE
00005184 7251                   <1>       JC     short DD_BAC       ; ASSUME NO DRIVE PRESENT
00005186 B530                   <1>       MOV    CH,TRK_SLAP        ; SEEK TO TRACK 48
00005188 E882FEFFFF             <1>       CALL   SEEK
0000518D 7248                   <1>       JC     short DD_BAC       ; ERROR NO DRIVE
0000518F B50B                   <1>       MOV    CH,QUIET_SEEK+1          ; SEEK TO TRACK 10
                                <1> SK_GIN:
00005191 FECD                   <1>       DEC    CH                 ; DECREMENT TO NEXT TRACK
00005193 6651                   <1>       PUSH   CX                 ; SAVE TRACK
00005195 E875FEFFFF             <1>       CALL   SEEK
0000519A 723C                   <1>       JC     short POP_BAC      ; POP AND RETURN
0000519C B8[D8510000]           <1>       MOV    eAX, POP_BAC       ; LOAD NEC OUTPUT ERROR ADDRESS
000051A1 50                     <1>       PUSH   eAX
000051A2 B404                   <1>       MOV    AH,SENSE_DRV_ST          ; SENSE DRIVE STATUS COMMAND BYTE
000051A4 E82CFEFFFF             <1>       CALL   NEC_OUTPUT         ; OUTPUT TO NEC
000051A9 6689F8                 <1>       MOV    AX,DI              ; AL = DRIVE
000051AC 88C4                   <1>       MOV    AH,AL              ; AH = DRIVE
000051AE E822FEFFFF             <1>       CALL   NEC_OUTPUT         ; OUTPUT TO NEC
000051B3 E85AFFFFFF             <1>       CALL   RESULTS                 ; GO GET STATUS
000051B8 58                     <1>       POP    eAX                ; THROW AWAY ERROR ADDRESS
000051B9 6659                   <1>       POP    CX                 ; RESTORE TRACK
000051BB F605[A18A0100]10       <1>       TEST   byte [NEC_STATUS], HOME    ; TRACK 0 ?
000051C2 74CD                   <1>       JZ     short SK_GIN       ; GO TILL TRACK 0
000051C4 08ED                   <1>       OR     CH,CH              ; IS HOME AT TRACK 0
000051C6 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>
000051C8 808F[AD8A0100]94       <1>       OR     byte [DSK_STATE+eDI], DRV_DET+MED_DET+RATE_250
000051CF C3                     <1>       RETn                      ; ALL INFORMATION SET
                                <1> IS_80:
000051D0 808F[AD8A0100]01       <1>       OR     byte [DSK_STATE+eDI], TRK_CAPA ; SETUP 80 TRACK CAPABILITY
                                <1> DD_BAC:
000051D7 C3                     <1>       RETn
                                <1> POP_BAC:
000051D8 6659                   <1>       POP    CX                 ; THROW AWAY
000051DA 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>
000051DB 6650                   <1>       PUSH   AX                 ; SAVE WORK REGISTER
000051DD 1E                     <1>       push   ds
000051DE 66B81000               <1>       mov    ax, KDATA
000051E2 8ED8                   <1>       mov    ds, ax
000051E4 800D[9D8A0100]80       <1>         OR      byte [SEEK_STATUS], INT_FLAG ; TURN ON INTERRUPT OCCURRED
000051EB B020                   <1>       MOV     AL,EOI                  ; END OF INTERRUPT MARKER
000051ED E620                   <1>       OUT    INTA00,AL          ; INTERRUPT CONTROL PORT
000051EF 1F                     <1>       pop    ds
000051F0 6658                   <1>       POP    AX                 ; RECOVER REGISTER
000051F2 CF                     <1>       IRETd                     ; RETURN FROM INTERRUPT
                                <1>
                                <1> ;--------------------------------------------------------------------------------
                                <1> ; DSKETTE_SETUP
                                <1> ;     THIS ROUTINE DOES A PRELIMINARY CHECK TO SEE WHAT TYPE OF
                                <1> ;     DISKETTE DRIVES ARE ATTACH TO THE SYSTEM.
                                <1> ;--------------------------------------------------------------------------------
                                <1>
                                <1> ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> DSKETTE_SETUP:
                                <1>       ;PUSH  AX                 ; SAVE REGISTERS
                                <1>       ;PUSH  BX
                                <1>       ;PUSH  CX
000051F3 52                     <1>       PUSH   eDX
                                <1>       ;PUSH  DI
                                <1>       ;;PUSH DS
                                <1>       ; 14/12/2014
                                <1>       ;mov   word [DISK_POINTER], MD_TBL6
                                <1>       ;mov   [DISK_POINTER+2], cs
                                <1>       ;
                                <1>       ;OR    byte [RTC_WAIT_FLAG], 1    ; NO RTC WAIT, FORCE USE OF LOOP
000051F4 31FF                   <1>       XOR    eDI,eDI                    ; INITIALIZE DRIVE POINTER
000051F6 66C705[AD8A0100]00-    <1>       MOV    WORD [DSK_STATE],0  ; INITIALIZE STATES
000051FE 00                     <1>
000051FF 8025[A88A0100]33       <1>       AND    byte [LASTRATE],~(STRT_MSK+SEND_MSK) ; CLEAR START & SEND
00005206 800D[A88A0100]C0       <1>       OR     byte [LASTRATE],SEND_MSK ; INITIALIZE SENT TO IMPOSSIBLE
0000520D C605[9D8A0100]00       <1>       MOV    byte [SEEK_STATUS],0       ; INDICATE RECALIBRATE NEEDED
00005214 C605[9F8A0100]00       <1>       MOV    byte [MOTOR_COUNT],0       ; INITIALIZE MOTOR COUNT
0000521B C605[9E8A0100]00       <1>       MOV    byte [MOTOR_STATUS],0      ; INITIALIZE DRIVES TO OFF STATE
00005222 C605[A08A0100]00       <1>       MOV    byte [DSKETTE_STATUS],0    ; NO ERRORS
                                <1>       ;
                                <1>       ; 28/02/2015
                                <1>       ;mov   word [cfd], 100h
00005229 E82BF2FFFF             <1>       call   DSK_RESET
```

```
3104 0000522E 5A                <1>       pop    edx
3105 0000522F F8                <1>       clc    ; 29/05/2016
3106 00005230 C3                <1>       retn
3107                            <1>
3108                            <1> ;SUP0:
3109                            <1> ;     CALL   DRIVE_DET           ; DETERMINE DRIVE
3110                            <1> ;     CALL   XLAT_OLD            ; TRANSLATE STATE TO COMPATIBLE MODE
3111                            <1> ;     ; 02/01/2015
3112                            <1> ;     ;INC   DI                  ; POINT TO NEXT DRIVE
3113                            <1> ;     ;CMP   DI,MAX_DRV          ; SEE IF DONE
3114                            <1> ;     ;JNZ   short SUP0          ; REPEAT FOR EACH ORIVE
3115                            <1> ;       cmp     byte [fd1_type], 0
3116                            <1> ;     jna    short sup1
3117                            <1> ;     or     di, di
3118                            <1> ;     jnz    short sup1
3119                            <1> ;     inc    di
3120                            <1> ;       jmp     short SUP0
3121                            <1> ;sup1:
3122                            <1> ;     MOV    byte [SEEK_STATUS],0       ; FORCE RECALIBRATE
3123                            <1> ;     ;AND   byte [RTC_WAIT_FLAG],0FEH ; ALLOW FOR RTC WAIT
3124                            <1> ;     CALL   SETUP_END           ; VARIOUS CLEANUPS
3125                            <1> ;     ;;POP  DS                  ; RESTORE CALLERS REGISTERS
3126                            <1> ;     ;POP   DI
3127                            <1> ;     POP    eDX
3128                            <1> ;     ;POP   CX
3129                            <1> ;     ;POP   BX
3130                            <1> ;     ;POP   AX
3131                            <1> ;     RETn
3132                            <1>
3133                            <1> ;////////////////////////////////////////////////////
3134                            <1> ;; END OF DISKETTE I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
3135                            <1> ;
3136                            <1>
3137                            <1> int13h: ; 21/02/2015
3138 00005231 F8                <1>       clc ; 20/07/2020
3139 00005232 9C                <1>       pushfd
3140 00005233 0E                <1>       push   cs
3141 00005234 E848010000        <1>       call   DISK_IO
3142 00005239 C3                <1>       retn
3143                            <1>
3144                            <1> ;;;;;; DISK I/O ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; 21/02/2015 ;;;
3145                            <1> ;////////////////////////////////////////////////////////////////////
3146                            <1>
3147                            <1> ; DISK I/O - Erdogan Tan (Retro UNIX 386 v1 project)
3148                            <1> ; 18/02/2016
3149                            <1> ; 17/02/2016
3150                            <1> ; 23/02/2015
3151                            <1> ; 21/02/2015 (unix386.s)
3152                            <1> ; 22/12/2014 - 14/02/2015 (dsectrm2.s)
3153                            <1> ;
3154                            <1> ; Original Source Code:
3155                            <1> ; DISK ----- 09/25/85 FIXED DISK BIOS
3156                            <1> ; (IBM PC XT Model 286 System BIOS Source Code, 04-21-86)
3157                            <1> ;
3158                            <1> ; Modifications: by reference of AWARD BIOS 1999 (D1A0622)
3159                            <1> ;             Source Code - ATORGS.ASM, AHDSK.ASM
3160                            <1> ;
3161                            <1>
3162                            <1>
3163                            <1> ;The wait for controller to be not busy is 10 seconds.
3164                            <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
3165                            <1> ;;WAIT_HDU_CTLR_BUSY_LO   equ    1615h
3166                            <1> ;;WAIT_HDU_CTLR_BUSY_HI   equ      05h
3167                            <1> WAIT_HDU_CTRL_BUSY_LH     equ    51615h  ;21/02/2015
3168                            <1>
3169                            <1> ;The wait for controller to issue completion interrupt is 10 seconds.
3170                            <1> ;10,000,000 / 30 = 333,333.  333,333 decimal = 051615h
3171                            <1> ;;WAIT_HDU_INT_LO  equ    1615h
3172                            <1> ;;WAIT_HDU_INT_HI  equ      05h
3173                            <1> WAIT_HDU_INT_LH           equ    51615h ; 21/02/2015
3174                            <1>
3175                            <1> ;The wait for Data request on read and write longs is
3176                            <1> ;2000 us. (?)
3177                            <1> ;;WAIT_HDU_DRQ_LO  equ    1000   ; 03E8h
3178                            <1> ;;WAIT_HDU_DRQ_HI  equ    0
3179                            <1> WAIT_HDU_DRQ_LH           equ    1000   ; 21/02/2015
3180                            <1>
3181                            <1> ; Port 61h (PORT_B)
3182                            <1> SYS1        equ    61h    ; PORT_B  (diskette.inc)
3183                            <1>
3184                            <1> ; 23/12/2014
3185                            <1> %define CMD_BLOCK       eBP-8  ; 21/02/2015
3186                            <1>
3187                            <1> ; 30/08/2020 - TRDOS 386 v2
3188                            <1>
3189                            <1> ;--- INT 13H -----------------------------------------------------------------
3190                            <1> ;                                                                        :
3191                            <1> ; FIXED DISK I/O INTERFACE                                                 :
3192                            <1> ;                                                                        :
3193                            <1> ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS THROUGH          :
3194                            <1> ;     THE IBM FIXED DISK CONTROLLER.                                     :
3195                            <1> ;                                                                        :
3196                            <1> ;     THE  BIOS  ROUTINES  ARE  MEANT  TO  BE  ACCESSED  THROUGH        :
3197                            <1> ;     SOFTWARE  INTERRUPTS  ONLY.    ANY  ADDRESSES  PRESENT     IN      :
3198                            <1> ;     THESE  LISTINGS  ARE  INCLUDED    ONLY  FOR  COMPLETENESS,        :
3199                            <1> ;     NOT  FOR  REFERENCE.  APPLICATIONS   WHICH  REFERENCE  ANY        :
3200                            <1> ;     ABSOLUTE  ADDRESSES  WITHIN  THE  CODE SEGMENTS  OF  BIOS        :
3201                            <1> ;     VIOLATE  THE  STRUCTURE  AND  DESIGN  OF  BIOS.                  :
3202                            <1> ;                                                                        :
3203                            <1> ;-------------------------------------------------------------------------:
3204                            <1> ;                                                                        :
3205                            <1> ; INPUT  (AH)= HEX COMMAND VALUE                                         :
3206                            <1> ;                                                                        :
3207                            <1> ;     (AH)= 00H  RESET DISK (DL = 80H,81H) / DISKETTE                    :
3208                            <1> ;     (AH)= 01H  READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)    :
```

```
3209                                    <1> ;               NOTE: DL < 80H - DISKETTE                                  :
3210                                    <1> ;                   DL > 80H - DISK                                :
3211                                    <1> ;     (AH)= 02H  READ THE DESIRED SECTORS INTO MEMORY                      :
3212                                    <1> ;     (AH)= 03H  WRITE THE DESIRED SECTORS FROM MEMORY                     :
3213                                    <1> ;     (AH)= 04H  VERIFY THE DESIRED SECTORS                                :
3214                                    <1> ;     (AH)= 05H  FORMAT THE DESIRED TRACK                                  :
3215                                    <1> ;     (AH)= 06H  UNUSED                                              :
3216                                    <1> ;     (AH)= 07H  UNUSED                                              :
3217                                    <1> ;     (AH)= 08H  RETURN THE CURRENT DRIVE PARAMETERS                           :
3218                                    <1> ;     (AH)= 09H  INITIALIZE DRIVE PAIR CHARACTERISTICS                     :
3219                                    <1> ;                INTERRUPT 41 POINTS TO DATA BLOCK FOR DRIVE 0           :
3220                                    <1> ;                INTERRUPT 46 POINTS TO DATA BLOCK FOR DRIVE 1           :
3221                                    <1> ;     (AH)= 0AH  READ LONG                                                 :
3222                                    <1> ;     (AH)= 0BH  WRITE LONG  (READ & WRITE LONG ENCOMPASS 512 + 4 BYTES ECC) :
3223                                    <1> ;     (AH)= 0CH  SEEK                                                :
3224                                    <1> ;     (AH)= 0DH  ALTERNATE DISK RESET (SEE DL)                             :
3225                                    <1> ;     (AH)= 0EH  UNUSED                                              :
3226                                    <1> ;     (AH)= 0FH  UNUSED                                              :
3227                                    <1> ;     (AH)= 10H  TEST DRIVE READY                                          :
3228                                    <1> ;     (AH)= 11H  RECALIBRATE                                               :
3229                                    <1> ;     (AH)= 12H  UNUSED                                              :
3230                                    <1> ;     (AH)= 13H  UNUSED                                              :
3231                                    <1> ;     (AH)= 14H  CONTROLLER INTERNAL DIAGNOSTIC                            :
3232                                    <1> ;     (AH)= 15H  READ DASD TYPE                                      :
3233                                    <1> ;                                                                    :
3234                                    <1> ;--------------------------------------------------------------------------
3235                                    <1> ;                                                                    :
3236                                    <1> ;     REGISTERS USED FOR FIXED DISK OPERATIONS                             :
3237                                    <1> ;                                                                    :
3238                                    <1> ;           (DL)   -  DRIVE NUMBER     (80H-81H FOR DISK. VALUE CHECKED)  :
3239                                    <1> ;           (DH)   -  HEAD NUMBER          (0-15 ALLOWED, NOT VALUE CHECKED)  :
3240                                    <1> ;           (CH)   -  CYLINDER NUMBER  (0-1023, NOT VALUE CHECKED)(SEE CL):
3241                                    <1> ;           (CL)   -  SECTOR NUMBER    (1-17, NOT VALUE CHECKED)         :
3242                                    <1> ;                                                                    :
3243                                    <1> ;                     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED     :
3244                                    <1> ;                          IN THE HIGH 2 BITS OF THE CL REGISTER         :
3245                                    <1> ;                          (10 BITS TOTAL)                               :
3246                                    <1> ;                                                                        :
3247                                    <1> ;           (AL)   -  NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,    :
3248                                    <1> ;                                       FOR READ/WRITE LONG 1-79H)       :
3249                                    <1> ;                                                                        :
3250                                    <1> ;           (ES:BX) -  ADDRESS OF BUFFER FOR READS AND WRITES,           :
3251                                    <1> ;                     (NOT REQUIRED FOR VERIFY)                          :
3252                                    <1> ;                                                                        :
3253                                    <1> ;           FORMAT (AH=5) ES:BX POINTS TO A 512 BYTE BUFFER. THE FIRST     :
3254                                    <1> ;                     2*(SECTORS/TRACK) BYTES CONTAIN F,N FOR EACH SECTOR.:
3255                                    <1> ;                     F = 00H FOR A GOOD SECTOR                          :
3256                                    <1> ;                         80H FOR A BAD SECTOR                           :
3257                                    <1> ;                     N = SECTOR NUMBER                                  :
3258                                    <1> ;                     FOR AN INTERLEAVE OF 2 AND 17 SECTORS/TRACK             :
3259                                    <1> ;                     THE TABLE SHOULD BE:                               :
3260                                    <1> ;                                                                        :
3261                                    <1> ;                DB     00H,01H,00H,0AH,00H,02H,00H,0BH,00H,03H,00H,0CH     :
3262                                    <1> ;                DB     00H,04H,00H,0DH,00H,05H,00H,0EH,00H,06H,00H,0FH     :
3263                                    <1> ;                DB     00H,07H,00H,10H,00H,08H,00H,11H,00H,09H             :
3264                                    <1> ;                                                                        :
3265                                    <1> ;--------------------------------------------------------------------------
3266                                    <1>
3267                                    <1> ;--------------------------------------------------------------------------
3268                                    <1> ; OUTPUT                                                             :
3269                                    <1> ;     AH = STATUS OF CURRENT OPERATION                               :
3270                                    <1> ;          STATUS BITS ARE DEFINED IN THE EQUATES BELOW                  :
3271                                    <1> ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)                       :
3272                                    <1> ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)                      :
3273                                    <1> ;                                                                        :
3274                                    <1> ;     NOTE:  ERROR 11H  INDICATES THAT THE DATA READ HAD A RECOVERABLE      :
3275                                    <1> ;            ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM.  THE DATA      :
3276                                    <1> ;            IS PROBABLY GOOD,   HOWEVER THE BIOS ROUTINE INDICATES AN      :
3277                                    <1> ;            ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE      :
3278                                    <1> ;            FOR ITSELF.  THE  ERROR  MAY  NOT  RECUR  IF  THE DATA IS      :
3279                                    <1> ;            REWRITTEN.                                                     :
3280                                    <1> ;                                                                           :
3281                                    <1> ;     IF DRIVE PARAMETERS WERE REQUESTED (DL >= 80H),                       :
3282                                    <1> ;        INPUT:                                                          :
3283                                    <1> ;          (DL) = DRIVE NUMBER                                           :
3284                                    <1> ;          ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)                                 :

3285                                    <1> ;          EBX = Buffer address for fixed disk parameters table (32 bytes)   :
3286                                    <1> ;        OUTPUT:                                                         :
3287                                    <1> ;          (DL) = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (1-2)  :
3288                                    <1> ;               (CONTROLLER CARD ZERO TALLY ONLY)                        :
3289                                    <1> ;          (DH) = MAXIMUM USEABLE VALUE FOR HEAD NUMBER                  :
3290                                    <1> ;          (CH) = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER                       :
3291                                    <1> ;          (CL) = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER                :
3292                                    <1> ;               AND CYLINDER NUMBER HIGH BITS                            :
3293                                    <1> ;                                                                        :
3294                                    <1> ;     IF READ DASD TYPE WAS REQUESTED,                                   :
3295                                    <1> ;                                                                        :
3296                                    <1> ;     AH = 0 - NOT PRESENT                                               :
3297                                    <1> ;          1 - DISKETTE - NO CHANGE LINE AVAILABLE                       :
3298                                    <1> ;          2 - DISKETTE - CHANGE LINE AVAILABLE                          :
3299                                    <1> ;          3 - FIXED DISK                                                :
3300                                    <1> ;                                                                        :
3301                                    <1> ;     CX,DX = NUMBER OF 512 BYTE BLOCKS WHEN AH = 3                      :
3302                                    <1> ;                                                                        :
3303                                    <1> ;     REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN        :
3304                                    <1> ;     INFORMATION.                                                       :
3305                                    <1> ;                                                                        :
3306                                    <1> ;     NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE        :
3307                                    <1> ;           ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.           :
3308                                    <1> ;                                                                        :
3309                                    <1> ;--------------------------------------------------------------------------
3310                                    <1>
3311                                    <1> SENSE_FAIL   EQU    0FFH         ; NOT IMPLEMENTED
3312                                    <1> NO_ERR       EQU    0E0H         ; STATUS ERROR/ERROR REGISTER=0
```

```
3313                    <1> WRITE_FAULT  EQU    0CCH          ; WRITE FAULT ON SELECTED DRIVE
3314                    <1> UNDEF_ERR    EQU    0BBH          ; UNDEFINED ERROR OCCURRED
3315                    <1> NOT_RDY      EQU    0AAH          ; DRIVE NOT READY
3316                    <1> TIME_OUT     EQU    80H           ; ATTACHMENT FAILED TO RESPOND
3317                    <1> BAD_SEEK     EQU    40H           ; SEEK OPERATION FAILED
3318                    <1> BAD_CNTLR    EQU    20H           ; CONTROLLER HAS FAILED
3319                    <1> DATA_CORRECTED     EQU    11H          ; ECC CORRECTED DATA ERROR
3320                    <1> BAD_ECC      EQU    10H           ; BAD ECC ON DISK READ
3321                    <1> BAD_TRACK    EQU    0BH           ; NOT IMPLEMENTED
3322                    <1> BAD_SECTOR   EQU    0AH           ; BAD SECTOR FLAG DETECTED
3323                    <1> ;DMA_BOUNDARY      EQU    09H          ; DATA EXTENDS TOO FAR
3324                    <1> INIT_FAIL    EQU    07H           ; DRIVE PARAMETER ACTIVITY FAILED
3325                    <1> BAD_RESET    EQU    05H           ; RESET FAILED
3326                    <1> ;RECORD_NOT_FND    EQU    04H          ; REQUESTED SECTOR NOT FOUND
3327                    <1> ;BAD_ADDR_MARK     EQU    02H          ; ADDRESS MARK NOT FOUND
3328                    <1> ;BAD_CMD     EQU    01H           ; BAD COMMAND PASSED TO DISK I/O
3329                    <1>
3330                    <1> ;--------------------------------------------------------
3331                    <1> ;                                                      :
3332                    <1> ; FIXED DISK PARAMETER TABLE                           :
3333                    <1> ;  -  THE TABLE IS COMPOSED OF A BLOCK DEFINED AS:     :
3334                    <1> ;                                                      :
3335                    <1> ;  +0 (1 WORD) - MAXIMUM NUMBER OF CYLINDERS           :
3336                    <1> ;  +2 (1 BYTE) - MAXIMUM NUMBER OF HEADS               :
3337                    <1> ;  +3 (1 WORD) - NOT USED/SEE PC-XT                    :
3338                    <1> ;  +5 (1 WORD) - STARTING WRITE PRECOMPENSATION CYL    :
3339                    <1> ;  +7 (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH         :
3340                    <1> ;  +8 (1 BYTE) - CONTROL BYTE                          :
3341                    <1> ;                BIT   7 DISABLE RETRIES -OR-          :
3342                    <1> ;                BIT   6 DISABLE RETRIES               :
3343                    <1> ;                BIT   3 MORE THAN 8 HEADS             :
3344                    <1> ;  +9 (3 BYTES)- NOT USED/SEE PC-XT                    :
3345                    <1> ; +12 (1 WORD) - LANDING ZONE                          :
3346                    <1> ; +14 (1 BYTE) - NUMBER OF SECTORS/TRACK               :
3347                    <1> ; +15 (1 BYTE) - RESERVED FOR FUTURE USE               :
3348                    <1> ;                                                      :
3349                    <1> ;      - TO DYNAMICALLY DEFINE A SET OF PARAMETERS     :
3350                    <1> ;        BUILD A TABLE FOR UP TO 15 TYPES AND PLACE    :
3351                    <1> ;        THE CORRESPONDING VECTOR INTO INTERRUPT 41    :
3352                    <1> ;        FOR DRIVE 0 AND INTERRUPT 46 FOR DRIVE 1.     :
3353                    <1> ;                                                      :
3354                    <1> ;--------------------------------------------------------
3355                    <1>
3356                    <1> ;--------------------------------------------------------
3357                    <1> ;                                                      :
3358                    <1> ; HARDWARE SPECIFIC VALUES                             :
3359                    <1> ;                                                      :
3360                    <1> ;  -  CONTROLLER I/O PORT                              :
3361                    <1> ;                                                      :
3362                    <1> ;     > WHEN READ FROM:                                :
3363                    <1> ;     HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU)          :
3364                    <1> ;     HF_PORT+1 - GET ERROR REGISTER                   :
3365                    <1> ;     HF_PORT+2 - GET SECTOR COUNT                     :
3366                    <1> ;     HF_PORT+3 - GET SECTOR NUMBER                    :
3367                    <1> ;     HF_PORT+4 - GET CYLINDER LOW                     :
3368                    <1> ;     HF_PORT+5 - GET CYLINDER HIGH (2 BITS)           :
3369                    <1> ;     HF_PORT+6 - GET SIZE/DRIVE/HEAD                  :
3370                    <1> ;     HF_PORT+7 - GET STATUS REGISTER                  :
3371                    <1> ;                                                      :
3372                    <1> ;     > WHEN WRITTEN TO:                               :
3373                    <1> ;     HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER)  :
3374                    <1> ;     HF_PORT+1 - SET PRECOMPENSATION CYLINDER         :
3375                    <1> ;     HF_PORT+2 - SET SECTOR COUNT                     :
3376                    <1> ;     HF_PORT+3 - SET SECTOR NUMBER                    :
3377                    <1> ;     HF_PORT+4 - SET CYLINDER LOW                     :
3378                    <1> ;     HF_PORT+5 - SET CYLINDER HIGH (2 BITS)           :
3379                    <1> ;     HF_PORT+6 - SET SIZE/DRIVE/HEAD                  :
3380                    <1> ;     HF_PORT+7 - SET COMMAND REGISTER                 :
3381                    <1> ;                                                      :
3382                    <1> ;--------------------------------------------------------
3383                    <1>
3384                    <1> ;HF_PORT     EQU    01F0H  ; DISK PORT
3385                    <1> ;HF1_PORT    equ    0170h
3386                    <1> ;HF_REG_PORT EQU    03F6H
3387                    <1> ;HF1_REG_PORT      equ    0376h
3388                    <1>
3389                    <1> HDC1_BASEPORT      equ    1F0h
3390                    <1> HDC2_BASEPORT      equ    170h
3391                    <1>
3392                    <1> align 2
3393                    <1>
3394                    <1> ;-----             STATUS REGISTER
3395                    <1>
3396                    <1> ST_ERROR     EQU    00000001B     ;
3397                    <1> ST_INDEX     EQU    00000010B     ;
3398                    <1> ST_CORRCTD   EQU    00000100B     ; ECC CORRECTION SUCCESSFUL
3399                    <1> ST_DRQ               EQU    00001000B     ;
3400                    <1> ST_SEEK_COMPL        EQU    00010000B     ; SEEK COMPLETE
3401                    <1> ST_WRT_FLT   EQU    00100000B     ; WRITE FAULT
3402                    <1> ST_READY     EQU    01000000B     ;
3403                    <1> ST_BUSY      EQU    10000000B     ;
3404                    <1>
3405                    <1> ;-----             ERROR REGISTER
3406                    <1>
3407                    <1> ERR_DAM      EQU    00000001B     ; DATA ADDRESS MARK NOT FOUND
3408                    <1> ERR_TRK_0    EQU    00000010B     ; TRACK 0 NOT FOUND ON RECAL
3409                    <1> ERR_ABORT    EQU    00000100B     ; ABORTED COMMAND
3410                    <1> ;            EQU    00001000B     ; NOT USED
3411                    <1> ERR_ID               EQU    00010000B     ; ID NOT FOUND
3412                    <1> ;            EQU    00100000B     ; NOT USED
3413                    <1> ERR_DATA_ECC EQU    01000000B
3414                    <1> ERR_BAD_BLOCK        EQU    10000000B
3415                    <1>
3416                    <1>
3417                    <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
0000523A C705[B88A0100]0000-    <1>       mov    dword [HDPM_TBL_VEC], (DPT_SEGM*16)+HD0_DPT
00005242 0900                   <1>
                                <1>       ;mov   word [HDPS_TBL_VEC],HD1_DPT      ; PARM TABLE DRIVE 81h
                                <1>       ;mov   word [HDPS_TBL_VEC+2],DPT_SEGM
00005244 C705[BC8A0100]2000-    <1>       mov    dword [HDPS_TBL_VEC], (DPT_SEGM*16)+HD1_DPT
0000524C 0900                   <1>
                                <1>       ;mov   word [HDSM_TBL_VEC],HD2_DPT      ; PARM TABLE DRIVE 82h
                                <1>       ;mov   word [HDSM_TBL_VEC+2],DPT_SEGM
0000524E C705[C08A0100]4000-    <1>       mov    dword [HDSM_TBL_VEC], (DPT_SEGM*16)+HD2_DPT
00005256 0900                   <1>
                                <1>       ;mov   word [HDSS_TBL_VEC],HD3_DPT      ; PARM TABLE DRIVE 83h
                                <1>       ;mov   word [HDSS_TBL_VEC+2],DPT_SEGM
00005258 C705[C48A0100]6000-    <1>       mov    dword [HDSS_TBL_VEC], (DPT_SEGM*16)+HD3_DPT
00005260 0900                   <1>
                                <1>       ;
                                <1>       ;;IN   AL,INTB01           ; TURN ON SECOND INTERRUPT CHIP
                                <1>       ;;;AND AL,0BFH
                                <1>       ;;and  al, 3Fh                   ; enable IRQ 14 and IRQ 15
                                <1>       ;;;JMP $+2
                                <1>       ;;IODELAY
                                <1>       ;;OUT  INTB01,AL
                                <1>       ;;IODELAY
                                <1>       ;;IN   AL,INTA01           ; LET INTERRUPTS PASS THRU TO
                                <1>       ;;AND  AL,0FBH             ;  SECOND CHIP
                                <1>       ;;;JMP $+2
                                <1>       ;;IODELAY
                                <1>       ;;OUT  INTA01,AL
```

```
                                <1>       ;
                                <1>       ;STI
                                <1>       ;;PUSH DS                 ; MOVE ABS0 POINTER TO
                                <1>       ;;POP  ES                 ; EXTRA SEGMENT POINTER
                                <1>       ;;;CALL      DDS                ; ESTABLISH DATA SEGMENT
                                <1>       ;;MOV  byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
                                <1>       ;;MOV  byte [HF_NUM],0           ; ZERO NUMBER OF FIXED DISKS
                                <1>       ;;MOV  byte [CONTROL_BYTE],0
                                <1>       ;;MOV  byte [PORT_OFF],0   ; ZERO CARD OFFSET
                                <1>       ; 20/12/2014 - private code by Erdogan Tan
                                <1>                    ; (out of original PC-AT, PC-XT BIOS code)
                                <1>       ;mov   si, hd0_type
00005262 BE[106E0000]           <1>       mov    esi, hd0_type
                                <1>       ;mov   cx, 4
00005267 B904000000             <1>       mov    ecx, 4
                                <1> hde_l:
0000526C AC                     <1>       lodsb
0000526D 3C80                   <1>       cmp    al, 80h                   ; 8?h = existing
0000526F 7206                   <1>       jb     short _L4
00005271 FE05[B48A0100]         <1>       inc    byte [HF_NUM]       ; + 1 hard (fixed) disk drives
                                <1> _L4: ; 26/02/2015
00005277 E2F3                   <1>       loop   hde_l
                                <1> ;_L4:                           ; 0 <= [HF_NUM] =< 4
                                <1> ;L4:
                                <1>       ;
                                <1>       ;; 31/12/2014 - cancel controller diagnostics here
                                <1>       ;;;mov       cx, 3  ; 26/12/2014 (Award BIOS 1999)
                                <1>       ;;mov  cl, 3
                                <1>       ;;
                                <1>       ;;MOV  DL,80H             ; CHECK THE CONTROLLER
                                <1> ;;hdc_dl:
                                <1>       ;;MOV  AH,14H             ; USE CONTROLLER DIAGNOSTIC COMMAND
                                <1>       ;;INT  13H                ; CALL BIOS WITH DIAGNOSTIC COMMAND
                                <1>       ;;;JC  short CTL_ERRX           ; DISPLAY ERROR MESSAGE IF BAD RETURN
                                <1>       ;;;jc  short POD_DONE ;22/12/2014
                                <1>       ;;jnc  short hdc_reset0
                                <1>       ;;loop hdc_dl
                                <1>       ;;; 27/12/2014
                                <1>       ;;stc
                                <1>       ;;retn
                                <1>       ;
                                <1> ;;hdc_reset0:
                                <1>       ; 18/01/2015
00005279 8A0D[B48A0100]         <1>       mov    cl, [HF_NUM]
0000527F 20C9                   <1>       and    cl, cl
00005281 740E                   <1>       jz     short POD_DONE
                                <1>       ;
00005283 B27F                   <1>       mov    dl, 7Fh
                                <1> hdc_reset1:
00005285 FEC2                   <1>       inc    dl
                                <1>       ;; 31/12/2015
                                <1>       ;;push dx
                                <1>       ;;push cx
                                <1>       ;;push ds
                                <1>       ;;sub  ax, ax
                                <1>       ;;mov  ds, ax
                                <1>       ;;MOV  AX, [TIMER_LOW]           ; GET START TIMER COUNTS
                                <1>       ;;pop  ds
                                <1>       ;;MOV  BX,AX
                                <1>       ;;ADD  AX,6*182           ; 60 SECONDS* 18.2
                                <1>       ;;MOV  CX,AX
                                <1>       ;;mov  word [wait_count], 0      ; 22/12/2014 (reset wait counter)
                                <1>       ;;
                                <1>       ;; 31/12/2014 - cancel HD_RESET_1
                                <1>       ;;CALL HD_RESET_1         ; SET UP DRIVE 0, (1,2,3)
                                <1>       ;;pop  cx
                                <1>       ;;pop  dx
                                <1>       ;;
                                <1>       ; 18/01/2015
00005287 B40D                   <1>       mov    ah, 0Dh ; ALTERNATE RESET
                                <1>       ;int   13h
00005289 E8A3FFFFFF             <1>       call   int13h
0000528E E2F5                   <1>       loop   hdc_reset1
00005290 F8                     <1>       clc    ; 29/05/2016
                                <1> POD_DONE:
00005291 C3                     <1>       RETn
                                <1>
                                <1> ;;-----      POD_ERROR
                                <1>
                                <1> ;;CTL_ERRX:
                                <1> ;     ;MOV   SI,OFFSET F1782     ; CONTROLLER ERROR
                                <1> ;     ;CALL  SET_FAIL            ; DO NOT IPL FROM DISK
                                <1> ;     ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
                                <1> ;     ;JMP   short POD_DONE
                                <1>
                                <1> ;;HD_RESET_1:
                                <1> ;;    ;PUSH  BX                  ; SAVE TIMER LIMITS
                                <1> ;;    ;PUSH  CX
                                <1> ;;RES_1: MOV AH,09H              ; SET DRIVE PARAMETERS
                                <1> ;;    INT    13H
                                <1> ;;    JC     short RES_2
                                <1> ;;    MOV    AH,11H              ; RECALIBRATE DRIVE
                                <1> ;;    INT    13H
                                <1> ;;    JNC    short RES_CK        ; DRIVE OK
                                <1> ;;RES_2: ;CALL     POD_TCHK           ; CHECK TIME OUT
                                <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
                                <1> ;;                                ; (30 seconds)
                                <1> ;;    ;cmc
                                <1> ;;    ;JNC   short RES_1
                                <1> ;;    jb     short RES_1
                                <1> ;;;RES_FL: ;MOV    SI,OFFSET F1781     ; INDICATE DISK 1 FAILURE;
                                <1> ;;    ;TEST  DL,1
                                <1> ;;    ;JNZ   RES_E1
                                <1> ;;    ;MOV   SI,OFFSET F1780     ; INDICATE DISK 0 FAILURE
                                <1> ;;    ;CALL  SET_FAIL            ; DO NOT TRY TO IPL DISK 0
```

```
                        <1> ;;    ;JMP   SHORT RES_E1
                        <1> ;;RES_ER: ; 22/12/2014
                        <1> ;;RES_OK:
                        <1> ;;    ;POP   CX                 ; RESTORE TIMER LIMITS
                        <1> ;;    ;POP   BX
                        <1> ;;    RETn
                        <1> ;;
                        <1> ;;RES_RS: MOV      AH,00H             ; RESET THE DRIVE
                        <1> ;;    INT    13H
                        <1> ;;RES_CK: MOV      AH,08H             ; GET MAX CYLINDER,HEAD,SECTOR
                        <1> ;;    MOV    BL,DL              ; SAVE DRIVE CODE
                        <1> ;;    INT    13H
                        <1> ;;    JC     short RES_ER
                        <1> ;;    MOV    [NEC_STATUS],CX    ; SAVE MAX CYLINDER, SECTOR
                        <1> ;;    MOV    DL,BL              ; RESTORE DRIVE CODE
                        <1> ;;RES_3: MOV AX,0401H           ; VERIFY THE LAST SECTOR
                        <1> ;;    INT    13H
                        <1> ;;    JNC    short RES_OK       ; VERIFY OK
                        <1> ;;    CMP    AH,BAD_SECTOR      ; OK ALSO IF JUST ID READ
                        <1> ;;    JE     short RES_OK
                        <1> ;;    CMP    AH,DATA_CORRECTED
                        <1> ;;    JE     short RES_OK
                        <1> ;;    CMP    AH,BAD_ECC
                        <1> ;;    JE     short RES_OK
                        <1> ;;    ;CALL  POD_TCHK           ; CHECK FOR TIME OUT
                        <1> ;;    cmp    word [wait_count], 6*182 ; waiting time (in timer ticks)
                        <1> ;;                               ; (60 seconds)
                        <1> ;;    cmc
                        <1> ;;    JC     short RES_ER       ; FAILED
                        <1> ;;    MOV    CX,[NEC_STATUS]    ; GET SECTOR ADDRESS, AND CYLINDER
                        <1> ;;    MOV    AL,CL              ; SEPARATE OUT SECTOR NUMBER
                        <1> ;;    AND    AL,3FH
                        <1> ;;    DEC    AL                 ; TRY PREVIOUS ONE
                        <1> ;;    JZ     short RES_RS       ; WE'VE TRIED ALL SECTORS ON TRACK
                        <1> ;;    AND    CL,0C0H            ; KEEP CYLINDER BITS
                        <1> ;;    OR     CL,AL              ; MERGE SECTOR WITH CYLINDER BITS
                        <1> ;;    MOV    [NEC_STATUS],CX    ; SAVE CYLINDER, NEW SECTOR NUMBER
                        <1> ;;    JMP    short RES_3        ; TRY AGAIN
                        <1> ;;;RES_ER: MOV     SI,OFFSET F1791     ; INDICATE DISK 1 ERROR
                        <1> ;;    ;TEST  DL,1
                        <1> ;;    ;JNZ   short RES_E1
                        <1> ;;    ;MOV   SI,OFFSET F1790     ; INDICATE DISK 0 ERROR
                        <1> ;;;RES_E1:
                        <1> ;;    ;CALL  E_MSG               ; DISPLAY ERROR AND SET (BP) ERROR FLAG
                        <1> ;;;RES_OK:
                        <1> ;;    ;POP   CX                  ; RESTORE TIMER LIMITS
                        <1> ;;    ;POP   BX
                        <1> ;;    ;RETn
                        <1> ;
                        <1> ;;SET_FAIL:
                        <1> ;     ;MOV   AX,X*(CMOS_DIAG+NMI)       ; GET CMOS ERROR BYTE
                        <1> ;     ;CALL  CMOS_READ
                        <1> ;     ;OR    AL,HF_FAIL          ; SET DO NOT IPL FROM DISK FLAG
                        <1> ;     ;XCHG  AH,AL               ; SAVE IT
                        <1> ;     ;CALL  CMOS_WRITE          ; PUT IT OUT
                        <1> ;     ;RETn
                        <1> ;
                        <1> ;;POD_TCHK:                      ; CHECK FOR 30 SECOND TIME OUT
                        <1> ;     ;POP   AX                  ; SAVE RETURN
                        <1> ;     ;POP   CX                  ; GET TIME OUT LIMITS
                        <1> ;     ;POP   BX
                        <1> ;     ;PUSH  BX                  ; AND SAVE THEM AGAIN
                        <1> ;     ;PUSH  CX
                        <1> ;     ;PUSH  AX
                        <1> ;     ;push  ds
                        <1> ;     ;xor   ax, ax
                        <1> ;     ;mov   ds, ax              ; RESTORE RETURN
                        <1> ;     ;MOV   AX, [TIMER_LOW]           ; AX = CURRENT TIME
                        <1> ;     ;                          ; BX = START TIME
                        <1> ;     ;                          ; CX = END TIME
                        <1> ;     ;pop   ds
                        <1> ;     ;CMP   BX,CX
                        <1> ;     ;JB    short TCHK1         ; START < END
                        <1> ;     ;CMP   BX,AX
                        <1> ;     ;JB    short TCHKG         ; END < START < CURRENT
                        <1> ;     ;JMP   SHORT TCHK2         ; END, CURRENT < START
                        <1> ;;TCHK1: CMP AX,BX
                        <1> ;;    JB     short TCHKNG        ; CURRENT < START < END
                        <1> ;;TCHK2: CMP AX,CX
                        <1> ;;    JB     short TCHKG         ; START < CURRENT < END
                        <1> ;;                               ; OR CURRENT < END < START
                        <1> ;;TCHKNG: STC                          ; CARRY SET INDICATES TIME OUT
                        <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> ; 30/08/2020
                        <1> ; 29/08/2020
                        <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
00005292 80642408FE     <1>       and   byte [esp+8], 11111110b  ; clear carry bit of eflags register
                        <1>       ; 16/05/2016
```

```
3729 00005297 1E                <1>       push   ds
3730 00005298 53                <1>       push   ebx ; user's buffer address (virtual)
3731 00005299 66BB1000          <1>       mov    bx, KDATA ; System (Kernel's) data segment
3732 0000529D 8EDB              <1>       mov    ds, bx
3733                            <1>
3734                            <1>       ;;15/01/2017
3735                            <1>       ; 14/01/2017
3736                            <1>       ; 02/01/2017
3737                            <1>       ;;mov  byte [intflg], 33h  ; disk io interrupt
3738                            <1>       ;pop   ebx
3739                            <1>       ;mov   [user_buffer], ebx
3740                            <1>
3741 0000529F 8F05[A4960100]   <1>       pop    dword [user_buffer] ; 01/06/2016
3742                            <1>
3743 000052A5 C605[DE8F0100]00  <1>       mov    byte [scount], 0 ; sector count for transfer
3744 000052AC 80FC03            <1>       cmp    ah, 03h ; chs write
3745 000052AF 7744              <1>       ja     short int33h_2
3746 000052B1 7407              <1>       je     short int33h_0
3747 000052B3 80FC02            <1>       cmp    ah, 02h ; chs read
3748 000052B6 726F              <1>       jb     short int33h_5
3749 000052B8 EB68              <1>       jmp    short int33h_4
3750                            <1> int33h_0:
3751                            <1>       ; transfer user's buffer content to sector buffer
3752 000052BA 51                <1>       push   ecx
3753 000052BB 0FB6C8            <1>       movzx  ecx, al
3754                            <1> int33h_1:
3755 000052BE 56                <1>       push   esi
3756 000052BF 8B35[A4960100]    <1>       mov    esi, [user_buffer]
3757                            <1>       ; esi = user's buffer address (virtual, ebx)
3758 000052C5 57                <1>       push   edi
3759 000052C6 06                <1>       push   es
3760 000052C7 50                <1>       push   eax
3761 000052C8 66B81000          <1>       mov    ax, KDATA
3762 000052CC 8EC0              <1>       mov    es, ax
3763 000052CE BF00000700        <1>       mov    edi, Cluster_Buffer
3764 000052D3 C1E109            <1>       shl    ecx, 9 ; * 512
3765 000052D6 E8B6C80000        <1>       call   transfer_from_user_buffer
3766 000052DB 58                <1>       pop    eax
3767 000052DC 07                <1>       pop    es
3768 000052DD 5F                <1>       pop    edi
3769 000052DE 5E                <1>       pop    esi
3770 000052DF 59                <1>       pop    ecx
3771 000052E0 7345              <1>       jnc    short int33h_5
3772 000052E2 8B1D[A4960100]    <1>       mov    ebx, [user_buffer] ; 01/06/2016
3773 000052E8 1F                <1>       pop    ds
3774                            <1>
3775                            <1>       ;;15/01/2017
3776                            <1>       ; 02/01/2017
3777                            <1>       ;cli
3778                            <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
3779                            <1>       ;
3780                            <1>       ; (*) 29/05/2016
3781                            <1>       ; (*) retf 4 ; skip eflags on stack
3782                            <1>
3783                            <1>       ; 29/05/2016 -set carry flag on stack-
3784                            <1>       ; [esp] = EIP
3785                            <1>       ; [esp+4] = CS
3786                            <1>       ; [esp+8] = E-FLAGS
3787 000052E9 804C240801        <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
3788                            <1>       ; [esp+12] = ESP (user)
3789                            <1>       ; [esp+16] = SS (User)
3790 000052EE B8FF000000        <1>       mov    eax, 0FFh ; Unknown error !?
3791                            <1>       ;iretd
3792 000052F3 EB7E              <1>       jmp    short int33h_7  ; 07/01/2017
3793                            <1>
3794                            <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
3795                            <1>       ; (OUTER-PRIVILEGE-LEVEL)
3796                            <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
3797                            <1>       ; // RETF instruction:
3798                            <1>       ;
3799                            <1>       ; IF OperandMode=32 THEN
3800                            <1>       ;    Load CS:EIP from stack;
3801                            <1>       ;    Set CS RPL to CPL;
3802                            <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
3803                            <1>       ;    Load SS:eSP from stack;
3804                            <1>       ; ELSE (* OperandMode=16 *)
3805                            <1>       ;    Load CS:IP from stack;
3806                            <1>       ;    Set CS RPL to CPL;
3807                            <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
3808                            <1>       ;    Load SS:eSP from stack;
3809                            <1>       ; FI;
3810                            <1>       ;
3811                            <1>       ; //
3812                            <1>
3813                            <1> int33h_2:
3814 000052F5 80FC05            <1>       cmp    ah, 05h ; format track
3815 000052F8 770A              <1>       ja     short int33h_3
3816 000052FA 722B              <1>       jb     short int33h_5
3817 000052FC 51                <1>       push   ecx
3818 000052FD B901000000        <1>       mov    ecx, 1
3819 00005302 EBBA              <1>       jmp    short int33h_1
3820                            <1> int33h_3:
3821 00005304 80FC1C            <1>       cmp    ah, 1Ch ; LBA write
3822 00005307 771E              <1>       ja     short int33h_5
3823 00005309 74AF              <1>       je     short int33h_0
3824 0000530B 80FC1B            <1>       cmp    ah, 1Bh ; LBA read
3825 0000530E 7412              <1>       je     short int33h_4
3826                            <1>       ; 29/08/2020
3827 00005310 80FC08            <1>       cmp    ah, 08h ; get disk parameters
3828                            <1>       ;jne   short int33h_5
3829 00005313 7405              <1>       je     short int33h_10
3830 00005315 80FC15            <1>       cmp    ah, 15h      ; read DASD type (get disk size)
3831 00005318 750D              <1>       jne    short int33h_5
3832                            <1> int33h_10:
3833                            <1>       ; 01/06/2016
```

```
3834 0000531A 8B1D[A4960100]      <1>       mov    ebx, [user_buffer] ; user's buffer address
3835 00005320 EB0A                <1>       jmp    short int33h_6
3836                              <1> int33h_4:
3837 00005322 A2[DE8F0100]        <1>       mov    byte [scount], al ; <= 128 sectors
3838                              <1> int33h_5:
3839 00005327 BB00000700          <1>       mov    ebx, Cluster_Buffer ; max. 65536 bytes
3840                              <1>                                  ; buf. addr: 70000h
3841                              <1>       ;mov   byte [ClusterBuffer_Valid], 0
3842                              <1> int33h_6:
3843 0000532C 1F                  <1>       pop    ds
3844 0000532D 9C                  <1>       pushfd
3845 0000532E 0E                  <1>       push   cs
3846 0000532F E84D000000          <1>       call   DISK_IO
3847 00005334 2E8B1D[A4960100]    <1>       mov    ebx, [CS:user_buffer] ; 01/06/2016
3848 0000533B 723D                <1>       jc     short int33h_9
3849                              <1>       ;
3850 0000533D 2E803D[DE8F0100]00  <1>       cmp    byte [CS:scount], 0
3851 00005345 762C                <1>       jna    short int33h_7
3852                              <1>       ;
3853                              <1>       ; transfer sector buffer content to user's buffer
3854 00005347 06                  <1>       push   es
3855 00005348 1E                  <1>       push   ds
3856 00005349 50                  <1>       push   eax
3857 0000534A 66B81000            <1>       mov    ax, KDATA
3858 0000534E 8ED8                <1>       mov    ds, ax
3859 00005350 8EC0                <1>       mov    es, ax
3860 00005352 51                  <1>       push   ecx
3861 00005353 56                  <1>       push   esi
3862 00005354 57                  <1>       push   edi
3863 00005355 0FB60D[DE8F0100]    <1>       movzx  ecx, byte [scount]
3864 0000535C C1E109              <1>       shl    ecx, 9 ; * 512 bytes
3865 0000535F 89DF                <1>       mov    edi, ebx ; user's buffer address
3866 00005361 BE00000700          <1>       mov    esi, Cluster_Buffer
3867 00005366 E8DCC70000          <1>       call   transfer_to_user_buffer
3868 0000536B 5F                  <1>       pop    edi
3869 0000536C 5E                  <1>       pop    esi
3870 0000536D 59                  <1>       pop    ecx
3871 0000536E 58                  <1>       pop    eax
3872 0000536F 1F                  <1>       pop    ds
3873 00005370 07                  <1>       pop    es
3874 00005371 7202                <1>       jc     short int33h_8
3875                              <1> int33h_7:
3876 00005373 FA                  <1>       cli
3877                              <1>       ;;15/01/2017
3878                              <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
3879                              <1>       ; cf = 0  ; use eflags which is in stack
3880 00005374 CF                  <1>       iretd
3881                              <1> int33h_8:
3882 00005375 B8FF000000          <1>       mov    eax, 0FFh ; Unknown error !?
3883                              <1> int33h_9:
3884                              <1>       ; cf = 1
3885                              <1>
3886                              <1>       ; (*) 29/05/2016
3887                              <1>       ; (*) retf 4 ; skip eflags on stack
3888                              <1>       ; Note: This 'retf 4' was wrong, -it was causing
3889                              <1>       ;       to stack errors in ring 3-
3890                              <1>       ;     POP sequence of 'retf 4' is as
3891                              <1>       ;       "eip, cs, eflags, esp, ss, +4 bytes"
3892                              <1>         ;       it is not as "eip, cs, +4 bytes, esp, ss" !
3893                              <1>
3894                              <1>       ; 29/05/2016 -set carry flag on stack-
3895 0000537A 804C240801          <1>       or     byte [esp+8], 1  ; set carry bit of eflags register
3896                              <1>       ;iretd
3897 0000537F EBF2                <1>       jmp    short int33h_7 ; 07/01/2017
3898                              <1>
3899                              <1> ; 30/08/2020
3900                              <1> ; 09/12/2017
3901                              <1> ; 29/05/2016
3902                              <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
3903                              <1>
3904                              <1> DISK_IO:
3905 00005381 80FA80              <1>       CMP    DL,80H             ; TEST FOR FIXED DISK DRIVE
3906                              <1>       ;JAE   short A1           ; YES, HANDLE HERE
3907                              <1>       ;;;INT 40H                ; DISKETTE HANDLER
3908                              <1>       ;;call int40h
3909 00005384 0F82FFEFFFFF        <1>       jb     DISKETTE_IO_1
3910                              <1> ;RET_2:
3911                              <1>       ;RETf  2                  ; BACK TO CALLER
3912                              <1> ;     retf   4
3913                              <1> A1:
3914 0000538A FB                  <1>       STI                       ; ENABLE INTERRUPTS
3915                              <1>       ;; 04/01/2015
3916                              <1>       ;;OR   AH,AH
3917                              <1>       ;;JNZ  short A2
3918                              <1>       ;;INT  40H                ; RESET NEC WHEN AH=0
3919                              <1>       ;;SUB  AH,AH
3920 0000538B 80FA83              <1>       CMP    DL,(80H + S_MAX_FILE - 1) ; 83h ; 30/08/2020
3921                              <1>       ;JA    short RET_2
3922 0000538E 7614                <1>       jna    short _A0
3923                              <1>       ; 29/05/2016
3924 00005390 1E                  <1>       push   ds
3925 00005391 50                  <1>       push   eax ; 30/08/2020 (ax --> eax)
3926 00005392 66B81000            <1>       mov    ax, KDATA
3927 00005396 8ED8                <1>       mov    ds, ax
3928 00005398 58                  <1>       pop    eax ;
3929 00005399 B4AA                <1>         mov     ah, 0AAh        ; Hard disk drive not ready !
3930                              <1>                                 ; (Programmer's guide to AMIBIOS, 1992)
3931 0000539B 8825[B38A0100]      <1>       mov     byte [DISK_STATUS1], ah
3932 000053A1 1F                  <1>       pop    ds
3933 000053A2 EB38                <1>       jmp    short RET_2
3934                              <1> _A0:
3935                              <1>       ; 18/01/2015
3936 000053A4 08E4                <1>       or     ah,ah
3937 000053A6 743A                <1>       jz     short A4
3938 000053A8 80FC0D              <1>       cmp    ah,0Dh ; Alternate reset
```

```
3939 000053AB 7504                <1>       jne    short A2
3940 000053AD 28E4                <1>       sub    ah,ah  ; Reset
3941 000053AF EB31                <1>       jmp    short A4
3942                              <1> A2:
3943 000053B1 80FC08              <1>       CMP    AH,08H              ; GET PARAMETERS IS A SPECIAL CASE
3944                              <1>       ;JNZ   short A3
3945                              <1>         ;JMP    GET_PARM_N
3946 000053B4 0F8452030000        <1>       je     GET_PARM_N
3947 000053BA 80FC15              <1> A3:   CMP    AH,15H              ; READ DASD TYPE IS ALSO
3948                              <1>       ;JNZ   short A4
3949                              <1>         ;JMP    READ_DASD_TYPE
3950 000053BD 0F84DB020000        <1>         je     READ_DASD_TYPE
3951                              <1>       ; 02/02/2015
3952 000053C3 80FC1D              <1>       cmp    ah, 1Dh                  ;(Temporary for Retro UNIX 386 v1)
3953                              <1>       ; 12/01/2015
3954 000053C6 F5                  <1>       cmc
3955 000053C7 7319                <1>       jnc    short A4
3956                              <1> int33h_bad_cmd:
3957                              <1>       ; 16/05/2016
3958                              <1>       ; 30/01/2015
3959                              <1>       ; 29/05/2016
3960 000053C9 1E                  <1>       push   ds
3961 000053CA 6650                <1>       push   ax
3962 000053CC 66B81000            <1>       mov    ax, KDATA
3963 000053D0 8ED8                <1>       mov    ds, ax
3964 000053D2 6658                <1>       pop    ax
3965 000053D4 B401                <1>       mov    ah, BAD_CMD
3966 000053D6 8825[B38A0100]      <1>       mov     [DISK_STATUS1], ah ; BAD_CMD  ; COMMAND ERROR
3967                              <1>         ;jmp short RET_2
3968                              <1> RET_2:
3969                              <1>       ; (*) 29/05/2016
3970                              <1>       ; (*) retf 4
3971 000053DC 804C240801          <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
3972 000053E1 CF                  <1>       iretd
3973                              <1> A4:                            ; SAVE REGISTERS DURING OPERATION
3974 000053E2 C8080000            <1>       ENTER  8,0               ; SAVE (BP) AND MAKE ROOM FOR @CMD_BLOCK
3975 000053E6 53                  <1>       PUSH   eBX               ;  IN THE STACK, THE COMMAND BLOCK IS:
3976 000053E7 51                  <1>       PUSH   eCX               ;   @CMD_BLOCK == BYTE PTR [BP]-8
3977 000053E8 52                  <1>       PUSH   eDX
3978 000053E9 1E                  <1>       PUSH   DS
3979 000053EA 06                  <1>       PUSH   ES
3980 000053EB 56                  <1>       PUSH   eSI
3981 000053EC 57                  <1>       PUSH   eDI
3982                              <1>       ;;04/01/2015
3983                              <1>       ;;OR   AH,AH             ; CHECK FOR RESET
3984                              <1>       ;;JNZ  short A5
3985                              <1>       ;;MOV  DL,80H            ; FORCE DRIVE 80 FOR RESET
3986                              <1> ;;A5:
3987                              <1>       ;push  cs
3988                              <1>       ;pop   ds
3989                              <1>       ; 21/02/2015
3990 000053ED 6650                <1>       push   ax
3991 000053EF 66B81000            <1>       mov    ax, KDATA
3992 000053F3 8ED8                <1>       mov    ds, ax
3993 000053F5 8EC0                <1>       mov    es, ax
3994 000053F7 6658                <1>       pop    ax
3995 000053F9 E88D000000          <1>       CALL   DISK_IO_CONT      ; PERFORM THE OPERATION
3996                              <1>       ;;CALL DDS               ; ESTABLISH SEGMENT
3997 000053FE 8A25[B38A0100]      <1>       MOV    AH,[DISK_STATUS1] ; GET STATUS FROM OPERATION
3998                              <1>       ;(*) CMP AH,1            ; SET THE CARRY FLAG TO INDICATE
3999                              <1>       ;(*) CMC                 ; SUCCESS OR FAILURE
4000 00005404 5F                  <1>       POP    eDI               ; RESTORE REGISTERS
4001 00005405 5E                  <1>       POP    eSI
4002 00005406 07                  <1>         POP     ES
4003 00005407 1F                  <1>         POP     DS
4004 00005408 5A                  <1>       POP    eDX
4005 00005409 59                  <1>       POP    eCX
4006 0000540A 5B                  <1>       POP    eBX
4007 0000540B C9                  <1>       LEAVE                    ; ADJUST (SP) AND RESTORE (BP)
4008                              <1>       ;RETf  2                 ; THROW AWAY SAVED FLAGS
4009                              <1>       ; (*) 29/05/2016
4010                              <1>       ; (*) retf 4
4011 0000540C 80FC01              <1>       cmp    ah, 1
4012 0000540F 7205                <1>       jc     short _A5
4013 00005411 804C240801          <1>       or     byte [esp+8], 1 ; set carry bit of eflags register
4014                              <1> _A5:
4015 00005416 CF                  <1>       iretd
4016                              <1>
4017                              <1> ; 21/02/2015
4018                              <1> ;       dw --> dd
4019                              <1> D1:                            ; FUNCTION TRANSFER TABLE
4020 00005417 [DA550000]          <1>       dd     DISK_RESET        ; 000H
4021 0000541B [51560000]          <1>       dd     RETURN_STATUS     ; 001H
4022 0000541F [5E560000]          <1>       dd     DISK_READ         ; 002H
4023 00005423 [67560000]          <1>       dd     DISK_WRITE        ; 003H
4024 00005427 [70560000]          <1>       dd     DISK_VERF         ; 004H
4025 0000542B [88560000]          <1>       dd     FMT_TRK           ; 005H
4026 0000542F [D0550000]          <1>       dd     BAD_COMMAND       ; 006H FORMAT BAD SECTORS
4027 00005433 [D0550000]          <1>       dd     BAD_COMMAND       ; 007H FORMAT DRIVE
4028 00005437 [D0550000]          <1>       dd     BAD_COMMAND       ; 008H RETURN PARAMETERS
4029 0000543B [A5570000]          <1>       dd     INIT_DRV          ; 009H
4030 0000543F [04580000]          <1>       dd     RD_LONG           ; 00AH
4031 00005443 [0D580000]          <1>       dd     WR_LONG           ; 00BH
4032 00005447 [16580000]          <1>       dd     DISK_SEEK         ; 00CH
4033 0000544B [DA550000]          <1>       dd     DISK_RESET        ; 00DH
4034 0000544F [D0550000]          <1>       dd     BAD_COMMAND       ; 00EH READ BUFFER
4035 00005453 [D0550000]          <1>       dd     BAD_COMMAND       ; 00FH WRITE BUFFER
4036 00005457 [3E580000]          <1>       dd     TST_RDY           ; 010H
4037 0000545B [62580000]          <1>       dd     HDISK_RECAL       ; 011H
4038 0000545F [D0550000]          <1>       dd     BAD_COMMAND       ; 012H MEMORY DIAGNOSTIC
4039 00005463 [D0550000]          <1>       dd     BAD_COMMAND       ; 013H DRIVE DIAGNOSTIC
4040 00005467 [98580000]          <1>       dd     CTLR_DIAGNOSTIC   ; 014H CONTROLLER DIAGNOSTIC
4041                              <1>       ; 02/02/2015 (Temporary - Retro UNIX 386 v1 - DISK I/O test)
4042 0000546B [D0550000]          <1>       dd     BAD_COMMAND       ; 015h
4043 0000546F [D0550000]          <1>       dd     BAD_COMMAND       ; 016h
```

```
4044 00005473 [D0550000]          <1>       dd     BAD_COMMAND         ; 017h
4045 00005477 [D0550000]          <1>       dd     BAD_COMMAND         ; 018h
4046 0000547B [D0550000]          <1>       dd     BAD_COMMAND         ; 019h
4047 0000547F [D0550000]          <1>       dd     BAD_COMMAND         ; 01Ah
4048 00005483 [5E560000]          <1>       dd     DISK_READ           ; 01Bh ; LBA read
4049 00005487 [67560000]          <1>       dd     DISK_WRITE          ; 01Ch ; LBA write
4050                              <1> D1L     EQU    $ - D1
4051                              <1>
4052                              <1> DISK_IO_CONT:
4053                              <1>       ;;CALL DDS                 ; ESTABLISH SEGMENT
4054 0000548B 80FC01              <1>       CMP    AH,01H              ; RETURN STATUS
4055                              <1>       ;;JNZ  short SU0
4056                              <1>         ;;JMP      RETURN_STATUS
4057 0000548E 0F84BD010000        <1>       je     RETURN_STATUS
4058                              <1> SU0:
4059 00005494 C605[B38A0100]00    <1>       MOV    byte [DISK_STATUS1],0     ; RESET THE STATUS INDICATOR
4060                              <1>       ;;PUSH BX                  ; SAVE DATA ADDRESS
4061                              <1>       ;mov   si, bx ;; 14/02/2015
4062 0000549B 89DE                <1>       mov    esi, ebx ; 21/02/2015
4063 0000549D 8A1D[B48A0100]      <1>       MOV    BL,[HF_NUM]         ; GET NUMBER OF DRIVES
4064                              <1>       ;; 04/01/2015
4065                              <1>       ;;PUSH AX
4066 000054A3 80E27F              <1>       AND    DL,7FH              ; GET DRIVE AS 0 OR 1
4067                              <1>                                  ; (get drive number as 0 to 3)
4068 000054A6 38D3                <1>       CMP    BL,DL
4069                              <1>         ;;JBE   BAD_COMMAND_POP         ; INVALID DRIVE
4070 000054A8 0F8622010000        <1>         jbe     BAD_COMMAND ;; 14/02/2015
4071                              <1>         ;
4072                              <1>       ;;03/01/2015
4073 000054AE 29DB                <1>       sub    ebx, ebx
4074 000054B0 88D3                <1>       mov    bl, dl
4075                              <1>       ;sub   bh, bh
4076 000054B2 883D[C88A0100]      <1>       mov    [LBAMode], bh       ; 0
4077                              <1>       ;;test byte [bx+hd0_type], 1     ; LBA ready ?
4078                              <1>       ;test  byte [ebx+hd0_type], 1
4079                              <1>       ;jz    short su1           ; no
4080                              <1>       ;inc   byte [LBAMode]
4081                              <1> ;su1:
4082                              <1>       ; 21/02/2015 (32 bit modification)
4083                              <1>       ;04/01/2015
4084 000054B8 6650                <1>       push   ax ; ***
4085                              <1>       ;PUSH  ES ; **
4086 000054BA 6652                <1>       PUSH   DX ; *
4087 000054BC 6650                <1>       push   ax
4088 000054BE E8BB060000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS
4089                              <1>       ; 02/02/2015
4090                              <1>       ;mov   ax, [ES:BX+16] ; I/O port base address (1F0h, 170h)
4091 000054C3 668B4310            <1>       mov    ax, [ebx+16]
4092 000054C7 66A3[006E0000]      <1>       mov    [HF_PORT], ax
4093                              <1>       ;mov   dx, [ES:BX+18] ; control port address (3F6h, 376h)
4094 000054CD 668B5312            <1>       mov    dx, [ebx+18]
4095 000054D1 668915[026E0000]    <1>       mov    [HF_REG_PORT], dx
4096                              <1>       ;mov   al, [ES:BX+20] ; head register upper nibble (A0h,B0h,E0h,F0h)
4097 000054D8 8A4314              <1>       mov    al, [ebx+20]
4098                              <1>       ; 23/02/2015
4099 000054DB A840                <1>       test   al, 40h       ; LBA bit (bit 6)
4100 000054DD 7406                <1>       jz     short su1
4101 000054DF FE05[C88A0100]      <1>       inc    byte [LBAMode] ; 1
4102                              <1> su1:
4103 000054E5 C0E804              <1>       shr    al, 4
4104 000054E8 2401                <1>       and    al, 1
4105 000054EA A2[046E0000]        <1>       mov    [hf_m_s], al
4106                              <1>       ;
4107                              <1>       ; 03/01/2015
4108                              <1>       ;MOV   AL,byte [ES:BX+8]   ; GET CONTROL BYTE MODIFIER
4109 000054EF 8A4308              <1>       mov    al, [ebx+8]
4110                              <1>       ;MOV   DX,[HF_REG_PORT]    ; Device Control register
4111 000054F2 EE                  <1>       OUT    DX,AL               ; SET EXTRA HEAD OPTION
4112                              <1>                                  ; Control Byte:  (= 08h, here)
4113                              <1>                                  ; bit 0 - 0
4114                              <1>                                  ; bit 1 - nIEN (1 = disable irq)
4115                              <1>                                  ; bit 2 - SRST (software RESET)
4116                              <1>                                  ; bit 3 - use extra heads (8 to 15)
4117                              <1>                                  ;         -always set to 1-
4118                              <1>                                  ; (bits 3 to 7 are reserved
4119                              <1>                                  ;          for ATA devices)
4120 000054F3 8A25[B58A0100]      <1>       MOV    AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
4121 000054F9 80E4C0              <1>       AND    AH,0C0H             ; CONTROL BYTE
4122 000054FC 08C4                <1>       OR     AH,AL
4123 000054FE 8825[B58A0100]      <1>       MOV    [CONTROL_BYTE],AH
4124                              <1>       ; 04/01/2015
4125 00005504 6658                <1>       pop    ax
4126 00005506 665A                <1>       pop    dx ; * ;; 14/02/2015
4127 00005508 20E4                <1>       and    ah, ah ; Reset function ?
4128 0000550A 7507                <1>       jnz    short su2
4129                              <1>       ;;pop  dx ; * ;; 14/02/2015
4130                              <1>       ;pop   es ; **
4131 0000550C 6658                <1>       pop    ax ; ***
4132                              <1>       ;;pop  bx
4133 0000550E E9C7000000          <1>         jmp     DISK_RESET
4134                              <1> su2:
4135 00005513 803D[C88A0100]00    <1>       cmp    byte [LBAMode], 0
4136 0000551A 7662                <1>       jna    short su3
4137                              <1>       ;
4138                              <1>       ; 02/02/2015 (LBA read/write function calls)
4139 0000551C 80FC1B              <1>       cmp    ah, 1Bh
4140 0000551F 720B                <1>       jb     short lbarw1
4141 00005521 80FC1C              <1>       cmp    ah, 1Ch
4142 00005524 775D                <1>       ja     short invldfnc
4143                              <1>       ;;pop  dx ; * ; 14/02/2015
4144                              <1>       ;mov   ax, cx ; Lower word of LBA address (bits 0-15)
4145 00005526 89C8                <1>       mov    eax, ecx ; LBA address (21/02/2015)
4146                              <1>       ;; 14/02/2015
4147 00005528 88D1                <1>       mov    cl, dl ; 14/02/2015
4148                              <1>       ;;mov  dx, bx
```

```
4149                                    <1>       ;mov   dx, si ; higher word of LBA address (bits 16-23)
4150                                    <1>       ;;mov  bx, di
4151                                    <1>       ;mov   si, di ; Buffer offset
4152 0000552A EB32                     <1>       jmp    short lbarw2
4153                                    <1> lbarw1:
4154                                    <1>       ; convert CHS to LBA
4155                                    <1>       ;
4156                                    <1>       ; LBA calculation - AWARD BIOS - 1999 - AHDSK.ASM
4157                                    <1>       ; LBA = "# of Heads" * Sectors/Track * Cylinder + Head * Sectors/Track
4158                                    <1>       ;      + Sector - 1
4159 0000552C 6652                     <1>       push   dx ; * ;; 14/02/2015
4160                                    <1>       ;xor   dh, dh
4161 0000552E 31D2                     <1>       xor    edx, edx
4162                                    <1>       ;mov   dl, [ES:BX+14]      ; sectors per track (logical)
4163 00005530 8A530E                   <1>       mov    dl, [ebx+14]
4164                                    <1>       ;xor   ah, ah
4165 00005533 31C0                     <1>       xor    eax, eax
4166                                    <1>       ;mov   al, [ES:BX+2]; heads (logical)
4167 00005535 8A4302                   <1>       mov    al, [ebx+2]
4168 00005538 FEC8                     <1>       dec    al
4169 0000553A 6640                     <1>       inc    ax           ; 0 =  256
4170 0000553C 66F7E2                   <1>       mul    dx
4171                                    <1>              ; AX = # of Heads" * Sectors/Track
4172 0000553F 6689CA                   <1>       mov    dx, cx
4173                                    <1>       ;and   cx, 3Fh       ; sector  (1 to 63)
4174 00005542 83E13F                   <1>       and    ecx, 3fh
4175 00005545 86D6                     <1>       xchg   dl, dh
4176 00005547 C0EE06                   <1>       shr    dh, 6
4177                                    <1>              ; DX = cylinder (0 to 1023)
4178                                    <1>       ;mul   dx
4179                                    <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder
4180 0000554A F7E2                     <1>       mul    edx
4181 0000554C FEC9                     <1>       dec    cl  ; sector - 1
4182                                    <1>       ;add   ax, cx
4183                                    <1>       ;adc   dx, 0
4184                                    <1>              ; DX:AX = # of Heads" * Sectors/Track * Cylinder + Sector -1
4185 0000554E 01C8                     <1>       add    eax, ecx
4186 00005550 6659                     <1>       pop    cx ; * ; ch = head, cl = drive number (zero based)
4187                                    <1>       ;push  dx
4188                                    <1>       ;push  ax
4189 00005552 50                       <1>       push   eax
4190                                    <1>       ;mov   al, [ES:BX+14]      ; sectors per track (logical)
4191 00005553 8A430E                   <1>       mov    al, [ebx+14]
4192 00005556 F6E5                     <1>       mul    ch
4193                                    <1>              ;  AX = Head * Sectors/Track
4194 00005558 0FB7C0                   <1>         movzx      eax, ax ; 09/12/2017
4195                                    <1>       ;pop   dx
4196 0000555B 5A                       <1>       pop    edx
4197                                    <1>       ;add   ax, dx
4198                                    <1>       ;pop   dx
4199                                    <1>       ;adc   dx, 0 ; add carry bit
4200 0000555C 01D0                     <1>       add    eax, edx
4201                                    <1> lbarw2:
4202 0000555E 29D2                     <1>       sub    edx, edx ; 21/02/2015
4203 00005560 88CA                     <1>       mov    dl, cl ; 21/02/2015
4204 00005562 C645F800                 <1>         mov     byte [CMD_BLOCK], 0 ; Features Register
4205                                    <1>                          ; NOTE: Features register (1F1h, 171h)
4206                                    <1>                          ; is not used for ATA device R/W functions.
4207                                    <1>                          ; It is old/obsolete 'write precompensation'
4208                                    <1>                          ; register and error register
4209                                    <1>                          ; for old ATA/IDE devices.
4210                                    <1>       ; 18/01/2014
4211                                    <1>       ;mov   ch, [hf_m_s] ; Drive 0 (master) or 1 (slave)
4212 00005566 8A0D[046E0000]           <1>       mov    cl, [hf_m_s]
4213                                    <1>       ;shl   ch, 4         ; bit 4 (drive bit)
4214                                    <1>       ;or    ch, 0E0h      ; bit 5 = 1
4215                                    <1>                          ; bit 6 = 1 = LBA mode
4216                                    <1>                          ; bit 7 = 1
4217 0000556C 80C90E                   <1>       or     cl, 0Eh ; 1110b
4218                                    <1>       ;and   dh, 0Fh             ; LBA byte 4 (bits 24 to 27)
4219 0000556F 25FFFFFF0F               <1>       and    eax, 0FFFFFFFh
4220 00005574 C1E11C                   <1>       shl    ecx, 28 ; 21/02/2015
4221                                    <1>       ;or    dh, ch
4222 00005577 09C8                     <1>       or     eax, ecx
4223                                    <1>       ;;mov  [CMD_BLOCK+2], al ; LBA byte 1 (bits 0 to 7)
4224                                    <1>                              ; (Sector Number Register)
4225                                    <1>       ;;mov  [CMD_BLOCK+3], ah ; LBA byte 2 (bits 8 to 15)
4226                                    <1>                              ; (Cylinder Low Register)
4227                                    <1>       ;mov   [CMD_BLOCK+2], ax ; LBA byte 1, 2
4228                                    <1>       ;mov   [CMD_BLOCK+4], dl ; LBA byte 3 (bits 16 to 23)
4229                                    <1>                              ; (Cylinder High Register)
4230                                    <1>       ;;mov  [CMD_BLOCK+5], dh ; LBA byte 4 (bits 24 to 27)
4231                                    <1>                              ; (Drive/Head Register)
4232                                    <1>
4233                                    <1>       ;mov   [CMD_BLOCK+4], dx ; LBA byte 4, LBA & DEV select bits
4234 00005579 8945FA                   <1>       mov    [CMD_BLOCK+2], eax ; 21/02/2015
4235                                    <1>       ;14/02/2015
4236                                    <1>       ;mov   dl, cl ; Drive number (INIT_DRV)
4237 0000557C EB38                     <1>       jmp    short su4
4238                                    <1> su3:
4239                                    <1>       ; 02/02/2015
4240                                    <1>       ; (Temporary functions 1Bh & 1Ch are not valid for CHS mode)
4241 0000557E 80FC14                   <1>       cmp    ah, 14h
4242 00005581 7604                     <1>       jna    short chsfnc
4243                                    <1> invldfnc:
4244                                    <1>         ; 14/02/2015
4245                                    <1>       ;pop   es ; **
4246 00005583 6658                     <1>         pop     ax ; ***
4247                                    <1>         ;jmp     short BAD_COMMAND_POP
4248 00005585 EB49                     <1>         jmp     short BAD_COMMAND
4249                                    <1> chsfnc:
4250                                    <1>       ;MOV   AX,[ES:BX+5]        ; GET WRITE PRE-COMPENSATION CYLINDER
4251 00005587 668B4305                 <1>       mov    ax, [ebx+5]
4252 0000558B 66C1E802                 <1>       SHR    AX,2
4253 0000558F 8845F8                   <1>       MOV    [CMD_BLOCK],AL
```

```
                        <1>       ;;MOV  AL,[ES:BX+8]        ; GET CONTROL BYTE MODIFIER
                        <1>       ;;PUSH DX
                        <1>       ;;MOV  DX,[HF_REG_PORT]
                        <1>       ;;OUT  DX,AL               ; SET EXTRA HEAD OPTION
                        <1>       ;;POP  DX ; *
                        <1>       ;;POP  ES ; **
                        <1>       ;;MOV  AH,[CONTROL_BYTE]   ; SET EXTRA HEAD OPTION IN
                        <1>       ;;AND  AH,0C0H             ; CONTROL BYTE
                        <1>       ;;OR   AH,AL
                        <1>       ;;MOV  [CONTROL_BYTE],AH
                        <1>       ;
00005592 88C8           <1>       MOV    AL,CL               ; GET SECTOR NUMBER
00005594 243F           <1>       AND    AL,3FH
00005596 8845FA         <1>       MOV    [CMD_BLOCK+2],AL
00005599 886DFB         <1>       MOV    [CMD_BLOCK+3],CH    ; GET CYLINDER NUMBER
0000559C 88C8           <1>       MOV    AL,CL
0000559E C0E806         <1>       SHR    AL,6
000055A1 8845FC         <1>       MOV    [CMD_BLOCK+4],AL    ; CYLINDER HIGH ORDER 2 BITS
                        <1>       ;;05/01/2015
                        <1>       ;;MOV  AL,DL               ; DRIVE NUMBER
000055A4 A0[046E0000]   <1>       mov    al, [hf_m_s]
000055A9 C0E004         <1>       SHL    AL,4
000055AC 80E60F         <1>       AND    DH,0FH              ; HEAD NUMBER
000055AF 08F0           <1>       OR     AL,DH
                        <1>       ;OR    AL,80H or 20H
000055B1 0CA0           <1>       OR     AL,80h+20h          ; ECC AND 512 BYTE SECTORS
000055B3 8845FD         <1>       MOV    [CMD_BLOCK+5],AL    ; ECC/SIZE/DRIVE/HEAD
                        <1> su4:
                        <1>       ;POP   ES ; **
                        <1>         ;; 14/02/2015
                        <1>         ;;POP   AX
                        <1>         ;;MOV   [CMD_BLOCK+1],AL        ; SECTOR COUNT
                        <1>         ;;PUSH  AX
                        <1>         ;;MOV   AL,AH                   ; GET INTO LOW BYTE
                        <1>         ;;XOR   AH,AH                   ; ZERO HIGH BYTE
                        <1>         ;;SAL   AX,1                    ; *2 FOR TABLE LOOKUP
000055B6 6658           <1>         pop     ax ; ***
000055B8 8845F9         <1>         mov     [CMD_BLOCK+1], al
000055BB 29DB           <1>         sub  ebx, ebx
000055BD 88E3           <1>       mov     bl, ah
                        <1>         ;xor     bh, bh
                        <1>         ;sal     bx, 1
000055BF 66C1E302       <1>         sal  bx, 2  ; 32 bit offset (21/02/2015)
                        <1>       ;;MOV   SI,AX                   ; PUT INTO SI FOR BRANCH
                        <1>         ;;CMP   AX,D1L                  ; TEST WITHIN RANGE
                        <1>         ;;JNB   short BAD_COMMAND_POP
                        <1>         ;cmp     bx, D1L
000055C3 83FB74         <1>       cmp    ebx, D1L
000055C6 7308           <1>       jnb    short BAD_COMMAND
                        <1>         ;xchg    bx, si
000055C8 87DE           <1>         xchg ebx, esi
                        <1>       ;;;POP AX                 ; RESTORE AX
                        <1>       ;;;POP BX                 ; AND DATA ADDRESS
                        <1>
                        <1>       ;;PUSH CX
                        <1>       ;;PUSH AX                 ; ADJUST ES:BX
                        <1>       ;MOV   CX,BX              ; GET 3 HIGH ORDER NIBBLES OF BX
                        <1>       ;SHR   CX,4
                        <1>       ;MOV   AX,ES
                        <1>       ;ADD   AX,CX
                        <1>       ;MOV   ES,AX
                        <1>       ;AND   BX,000FH           ; ES:BX CHANGED TO ES:000X
                        <1>       ;;POP  AX
                        <1>       ;;POP  CX
                        <1>       ;;JMP  word [CS:SI+D1]
                        <1>       ;jmp   word [SI+D1]
000055CA FFA6[17540000] <1>       jmp    dword [esi+D1]
                        <1> ;;BAD_COMMAND_POP:
                        <1> ;;    POP    AX
                        <1> ;;    POP    BX
                        <1> BAD_COMMAND:
000055D0 C605[B38A0100]01 <1>         MOV     byte [DISK_STATUS1],BAD_CMD  ; COMMAND ERROR
000055D7 B000           <1>       MOV    AL,0
000055D9 C3             <1>       RETn
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     RESET THE DISK SYSTEM  (AH=00H) :
                        <1> ;----------------------------------------
                        <1>
                        <1> ; 18-1-2015 : one controller reset (not other one)
                        <1>
                        <1> DISK_RESET:
000055DA FA             <1>       CLI
000055DB E4A1           <1>       IN     AL,INTB01          ; GET THE MASK REGISTER
                        <1>       ;JMP   $+2
                        <1>       IODELAY
000055DD EB00           <2>  jmp short $+2
000055DF EB00           <2>  jmp short $+2
                        <1>       ;AND   AL,0BFH             ; ENABLE FIXED DISK INTERRUPT
000055E1 243F           <1>       and    al,3Fh             ; 22/12/2014 (IRQ 14 & IRQ 15)
000055E3 E6A1           <1>       OUT    INTB01,AL
000055E5 FB             <1>       STI                       ; START INTERRUPTS
                        <1>       ; 14/02/2015
000055E6 6689D7         <1>       mov    di, dx
                        <1>       ; 04/01/2015
                        <1>       ;xor   di,di
                        <1> drst0:
000055E9 B004           <1>       MOV    AL,04H  ; bit 2 - SRST
                        <1>       ;MOV   DX,HF_REG_PORT
000055EB 668B15[026E0000] <1>     MOV    DX,[HF_REG_PORT]
000055F2 EE             <1>       OUT    DX,AL              ; RESET
                        <1> ;     MOV    CX,10              ; DELAY COUNT
                        <1> ;DRD: DEC    CX
                        <1> ;     JNZ    short DRD          ; WAIT 4.8 MICRO-SEC
                        <1>       ;mov   cx,2               ; wait for 30 micro seconds
```

```
4357 000055F3 B902000000          <1>         mov  ecx, 2 ; 21/02/2015
4358 000055F8 E851CEFFFF          <1>       call    WAITF                   ; (Award Bios 1999 - WAIT_REFRESH,
4359                              <1>                                         ; 40 micro seconds)
4360 000055FD A0[B58A0100]        <1>       mov    al,[CONTROL_BYTE]
4361 00005602 240F                <1>       AND    AL,0FH              ; SET HEAD OPTION
4362 00005604 EE                  <1>       OUT    DX,AL               ; TURN RESET OFF
4363 00005605 E86A040000          <1>       CALL   NOT_BUSY
4364 0000560A 7515                <1>       JNZ    short DRERR         ; TIME OUT ON RESET
4365 0000560C 668B15[006E0000]    <1>       MOV    DX,[HF_PORT]
4366 00005613 FEC2                <1>       inc    dl  ; HF_PORT+1
4367                              <1>       ; 02/01/2015 - Award BIOS 1999 - AHDSK.ASM
4368                              <1>         ;mov     cl, 10
4369 00005615 B90A000000          <1>         mov     ecx, 10 ; 21/02/2015
4370                              <1> drst1:
4371 0000561A EC                  <1>       IN     AL,DX               ; GET RESET STATUS
4372 0000561B 3C01                <1>       CMP    AL,1
4373                              <1>       ; 04/01/2015
4374 0000561D 740A                <1>       jz     short drst2
4375                              <1>       ;JNZ   short DRERR         ; BAD RESET STATUS
4376                              <1>              ; Drive/Head Register - bit 4
4377 0000561F E2F9                <1>       loop   drst1
4378                              <1> DRERR:
4379 00005621 C605[B38A0100]05    <1>       MOV    byte [DISK_STATUS1],BAD_RESET ; CARD FAILED
4380 00005628 C3                  <1>       RETn
4381                              <1> drst2:
4382                              <1>       ; 14/02/2015
4383 00005629 6689FA              <1>       mov    dx,di
4384                              <1> ;drst3:
4385                              <1> ;     ; 05/01/2015
4386                              <1> ;     shl    di,1
4387                              <1> ;     ; 04/01/2015
4388                              <1> ;     mov    ax,[di+hd_cports]
4389                              <1> ;     cmp    ax,[HF_REG_PORT]
4390                              <1> ;     je     short drst4
4391                              <1> ;     mov    [HF_REG_PORT], ax
4392                              <1> ;     ; 03/01/2015
4393                              <1> ;     mov    ax,[di+hd_ports]
4394                              <1> ;       mov     [HF_PORT], ax
4395                              <1> ;     ; 05/01/2014
4396                              <1> ;     shr    di,1
4397                              <1> ;     ; 04/01/2015
4398                              <1> ;     jmp    short drst0  ; reset other controller
4399                              <1> ;drst4:
4400                              <1> ;     ; 05/01/2015
4401                              <1> ;     shr    di,1
4402                              <1> ;     mov    al,[di+hd_dregs]
4403                              <1> ;     and    al,10h ; bit 4 only
4404                              <1> ;     shr    al,4 ; bit 4  -> bit 0
4405                              <1> ;     mov    [hf_m_s], al ; (0 = master, 1 = slave)
4406                              <1>       ;
4407 0000562C A0[046E0000]        <1>       mov    al, [hf_m_s] ; 18/01/2015
4408 00005631 A801                <1>       test   al,1
4409                              <1> ;     jnz    short drst6
4410 00005633 7516                <1>         jnz     short drst4
4411 00005635 8065FDEF            <1>       AND     byte [CMD_BLOCK+5],0EFH ; SET TO DRIVE 0
4412                              <1> ;drst5:
4413                              <1> drst3:
4414 00005639 E867010000          <1>       CALL   INIT_DRV            ; SET MAX HEADS
4415                              <1>       ;mov   dx,di
4416 0000563E E81F020000          <1>       CALL   HDISK_RECAL         ; RECAL TO RESET SEEK SPEED
4417                              <1>       ; 04/01/2014
4418                              <1> ;     inc    di
4419                              <1> ;     mov    dx,di
4420                              <1> ;     cmp    dl,[HF_NUM]
4421                              <1> ;     jb     short drst3
4422                              <1> ;DRE:
4423 00005643 C605[B38A0100]00    <1>       MOV    byte [DISK_STATUS1],0     ; IGNORE ANY SET UP ERRORS
4424 0000564A C3                  <1>       RETn
4425                              <1> ;drst6:
4426                              <1> drst4:             ; Drive/Head Register - bit 4
4427 0000564B 804DFD10            <1>       OR      byte [CMD_BLOCK+5],010H ; SET TO DRIVE 1
4428                              <1>         ;jmp    short drst5
4429 0000564F EBE8                <1>         jmp     short drst3
4430                              <1>
4431                              <1> ;--------------------------------------
4432                              <1> ;     DISK STATUS ROUTINE  (AH = 01H) :
4433                              <1> ;--------------------------------------
4434                              <1>
4435                              <1> RETURN_STATUS:
4436 00005651 A0[B38A0100]        <1>       MOV    AL,[DISK_STATUS1]   ; OBTAIN PREVIOUS STATUS
4437 00005656 C605[B38A0100]00    <1>         MOV     byte [DISK_STATUS1],0   ; RESET STATUS
4438 0000565D C3                  <1>       RETn
4439                              <1>
4440                              <1> ;--------------------------------------
4441                              <1> ;     DISK READ ROUTINE    (AH = 02H) :
4442                              <1> ;--------------------------------------
4443                              <1>
4444                              <1> DISK_READ:
4445 0000565E C645FE20            <1>       MOV    byte [CMD_BLOCK+6],READ_CMD
4446 00005662 E986020000          <1>         JMP     COMMANDI
4447                              <1>
4448                              <1> ;--------------------------------------
4449                              <1> ;     DISK WRITE ROUTINE   (AH = 03H) :
4450                              <1> ;--------------------------------------
4451                              <1>
4452                              <1> DISK_WRITE:
4453 00005667 C645FE30            <1>       MOV    byte [CMD_BLOCK+6],WRITE_CMD
4454 0000566B E9D8020000          <1>         JMP     COMMANDO
4455                              <1>
4456                              <1> ;--------------------------------------
4457                              <1> ;     DISK VERIFY       (AH = 04H) :
4458                              <1> ;--------------------------------------
4459                              <1>
4460                              <1> DISK_VERF:
4461 00005670 C645FE40            <1>       MOV    byte [CMD_BLOCK+6],VERIFY_CMD
```

```
4462 00005674 E846030000          <1>       CALL   COMMAND
4463 00005679 750C                <1>       JNZ    short VERF_EXIT           ; CONTROLLER STILL BUSY
4464 0000567B E8B8030000          <1>       CALL   _WAIT               ; (Original: CALL WAIT)
4465 00005680 7505                <1>       JNZ    short VERF_EXIT           ; TIME OUT
4466 00005682 E845040000          <1>       CALL   CHECK_STATUS
4467                              <1> VERF_EXIT:
4468 00005687 C3                  <1>       RETn
4469                              <1>
4470                              <1> ;---------------------------------------
4471                              <1> ;     FORMATTING        (AH = 05H) :
4472                              <1> ;---------------------------------------
4473                              <1>
4474                              <1> FMT_TRK:                        ; FORMAT TRACK      (AH = 005H)
4475 00005688 C645FE50            <1>       MOV    byte [CMD_BLOCK+6],FMTTRK_CMD
4476                              <1>       ;PUSH  ES
4477                              <1>       ;PUSH  BX
4478 0000568C 53                  <1>       push   ebx
4479 0000568D E8EC040000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS ADDRESS
4480                              <1>       ;MOV   AL,[ES:BX+14]       ; GET SECTORS/TRACK
4481 00005692 8A430E              <1>       mov    al, [ebx+14]
4482 00005695 8845F9              <1>       MOV    [CMD_BLOCK+1],AL    ; SET SECTOR COUNT IN COMMAND
4483 00005698 5B                  <1>       pop    ebx
4484                              <1>       ;POP   BX
4485                              <1>       ;POP   ES
4486 00005699 E9B1020000          <1>         JMP     CMD_OF                  ; GO EXECUTE THE COMMAND
4487                              <1>
4488                              <1> ; 30/08/2020
4489                              <1>
4490                              <1> ;---------------------------------------
4491                              <1> ;     READ DASD TYPE           (AH = 15H) :
4492                              <1> ;---------------------------------------
4493                              <1>
4494                              <1> READ_DASD_TYPE:
4495                              <1> READ_D_T:                       ; GET DRIVE PARAMETERS
4496 0000569E 1E                  <1>       PUSH   DS                 ; SAVE REGISTERS
4497                              <1>       ;PUSH  ES
4498 0000569F 53                  <1>       PUSH   eBX
4499                              <1>       ;CALL  DDS                ; ESTABLISH ADDRESSING
4500                              <1>       ;push  cs
4501                              <1>       ;pop   ds
4502 000056A0 66BB1000            <1>         mov  bx, KDATA
4503 000056A4 8EDB                <1>       mov    ds, bx
4504                              <1>       ;mov   es, bx
4505 000056A6 C605[B38A0100]00    <1>       MOV     byte [DISK_STATUS1],0
4506 000056AD 8A1D[B48A0100]      <1>       MOV    BL,[HF_NUM]         ; GET NUMBER OF DRIVES
4507 000056B3 80E27F              <1>       AND    DL,7FH              ; GET DRIVE NUMBER
4508 000056B6 38D3                <1>       CMP    BL,DL
4509 000056B8 763C                <1>       JBE    short RDT_NOT_PRESENT     ; RETURN DRIVE NOT PRESENT
4510 000056BA 0FB6C2              <1>       movzx  eax, dl ; 28/08/2020
4511 000056BD E8BC040000          <1>       CALL   GET_VEC             ; GET DISK PARAMETERS ADDRESS
4512                              <1>
4513                              <1>       ; 28/08/2020 - TRDOS 386 v2
4514 000056C2 F6431440            <1>       test   byte [ebx+20], 40h  ; LBA bit (bit 6)
4515 000056C6 751D                <1>       jnz    short RDT3 ; LBA disk (may be > 8GB)
4516                              <1>
4517                              <1>       ;MOV   AL,[ES:BX+2]        ; HEADS
4518 000056C8 8A4302              <1>       mov    al, [ebx+2]
4519                              <1>       ;MOV   CL,[ES:BX+14]
4520 000056CB 8A4B0E              <1>       mov    cl, [ebx+14]
4521 000056CE F6E9                <1>       IMUL   CL                 ; * NUMBER OF SECTORS
4522                              <1>       ;MOV   CX,[ES:BX]          ; MAX NUMBER OF CYLINDERS
4523 000056D0 668B0B              <1>       mov    cx, [ebx]
4524                              <1>       ;
4525                              <1>       ; 02/01/2015
4526                              <1>       ; ** leave the last cylinder as reserved for diagnostics **
4527                              <1>       ; (Also in Award BIOS - 1999, AHDSK.ASM, FUN15 -> sub ax, 1)
4528 000056D3 6649                <1>       DEC    CX                 ; LEAVE ONE FOR DIAGNOSTICS
4529                              <1>       ;
4530 000056D5 66F7E9              <1>       IMUL   CX                 ; NUMBER OF SECTORS
4531 000056D8 6689D1              <1>       MOV    CX,DX              ; HIGH ORDER HALF
4532 000056DB 6689C2              <1>       MOV    DX,AX              ; LOW ORDER HALF
4533                              <1>       ;SUB   AX,AX
4534                              <1>       ; 28/08/2020
4535                              <1>       ;sub   al, al
4536                              <1> RDT4:
4537 000056DE 29C0                <1>       sub    eax, eax ; 28/08/2020
4538                              <1>       ;
4539 000056E0 B403                <1>       MOV    AH,03H             ; INDICATE FIXED DISK
4540                              <1>       ; 30/08/2020 (clc is not needed here)
4541                              <1>       ;and   byte [esp+8], 0FEh ; clear carry bit of eflags register
4542                              <1> RDT2:
4543 000056E2 5B                  <1>       POP    eBX                ; RESTORE REGISTERS
4544                              <1>       ;POP   ES
4545 000056E3 1F                  <1>       POP    DS
4546                              <1>       ; (*) CLC                 ; CLEAR CARRY
4547                              <1>       ;RETf  2
4548                              <1>       ; (*) 29/05/2016
4549                              <1>       ; (*) retf 4
4550                              <1>       ; 30/08/2020 (clc is not needed here)
4551                              <1>       ;and   byte [esp+8], 0FEh ; clear carry bit of eflags register
4552 000056E4 CF                  <1>       iretd
4553                              <1>
4554                              <1> RDT3: ; 28/08/2020
4555                              <1>       ; (use the result of INT 13h, function 48h as disk size)
4556                              <1>       ; eax = al = zero based hard disk number
4557                              <1>       ; 29/08/2020
4558                              <1>       ;add   al, 2 ; hd0 = physical disk drive 2
4559 000056E5 C0E002              <1>       shl    al, 2 ; * 4
4560                              <1> RDT5:
4561                              <1>       ;add   eax, drv.size
4562 000056E8 05[466E0000]        <1>       add    eax, drv.size+8 ; 29/08/2020
4563 000056ED 668B10              <1>       mov    dx, [eax]   ; low word of disk size
4564 000056F0 668B4802            <1>       mov    cx, [eax+2] ; high word of disk size
4565                              <1>       ;sub   eax, eax
4566 000056F4 EBE8                <1>       jmp    short RDT4
```

```
                                <1>
                                <1> RDT_NOT_PRESENT:
                                <1>       ; 30/08/2020
000056F6 C605[B38A0100]AA       <1>       mov   byte [DISK_STATUS1], NOT_RDY ; DRIVE NOT READY
                                <1>
                                <1>       ;SUB   AX,AX              ; DRIVE NOT PRESENT RETURN
000056FD 29C0                   <1>       sub   eax, eax ; 30/08/2020
000056FF 6689C1                 <1>       MOV   CX,AX               ; ZERO BLOCK COUNT
00005702 6689C2                 <1>       MOV   DX,AX
                                <1>       ; 30/08/2020
00005705 804C241001             <1>       or    byte [esp+16], 1 ; set carry bit of eflags register
                                <1>             ; cf = 1 -> ah = 0, drive not ready, disk type = 0
0000570A EBD6                   <1>       JMP   short RDT2
                                <1>
                                <1> ; 28/05/2016
                                <1> ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> ;----------------------------------------
                                <1> ;     GET PARAMETERS           (AH = 08H) :
                                <1> ;----------------------------------------
                                <1>
                                <1> GET_PARM_N:
                                <1>       ; ebx = user's buffer address for parameters table
                                <1> ;GET_PARM:                    ; GET DRIVE PARAMETERS
0000570C 1E                     <1>       PUSH  DS                ; SAVE REGISTERS
0000570D 06                     <1>       PUSH  ES
0000570E 53                     <1>       PUSH  eBX
                                <1>       ;MOV  AX,ABS0           ; ESTABLISH ADDRESSING
                                <1>       ;MOV  DS,AX
                                <1>       ;TEST DL,1              ; CHECK FOR DRIVE 1
                                <1>       ;JZ   short G0
                                <1>       ;LES  BX,@HF1_TBL_VEC
                                <1>       ;JMP  SHORT G1
                                <1> ;G0:  LES   BX,@HF_TBL_VEC
                                <1> ;G1:
                                <1>       ;CALL DDS               ; ESTABLISH SEGMENT
                                <1>       ; 22/12/2014
                                <1>       ;push cs
                                <1>       ;pop  ds
0000570F 66BB1000               <1>       mov   bx, KDATA
00005713 8EDB                   <1>       mov   ds, bx
00005715 8EC3                   <1>       mov   es, bx ; 27/05/2016
                                <1>       ;
00005717 80EA80                 <1>       SUB   DL,80H
                                <1>       ;CMP  DL,MAX_FILE         ; TEST WITHIN RANGE
                                <1>       ;JAE  short G4 ; 29/08/2020 - BugFix
                                <1>       ; 30/08/2020
0000571A 3A15[B48A0100]         <1>       cmp   dl, [HF_NUM] ; is hard disk index < hard disk count ?
00005720 736A                   <1>       jae   short G4     ; no, error ! drive not ready !
                                <1>       ;
00005722 31DB                   <1>       xor   ebx, ebx ; 21/02/2015
                                <1>       ; 22/12/2014
00005724 88D3                   <1>       mov   bl, dl
                                <1>       ;xor  bh, bh
00005726 C0E302                 <1>       shl   bl, 2               ; convert index to offset
                                <1>       ;add  bx, HF_TBL_VEC
00005729 81C3[B88A0100]         <1>       add   ebx, HF_TBL_VEC
                                <1>       ;mov  ax, [bx+2]
                                <1>       ;mov  es, ax              ; dpt segment
                                <1>       ;mov  bx, [bx]            ; dpt offset
0000572F 8B1B                   <1>       mov   ebx, [ebx] ; 32 bit offset
                                <1>
00005731 C605[B38A0100]00       <1>       MOV   byte [DISK_STATUS1],0
                                <1>         ;MOV AX,[ES:BX]         ; MAX NUMBER OF CYLINDERS
00005738 668B03                 <1>       mov   ax, [ebx]
                                <1>       ;;SUB  AX,2               ; ADJUST FOR 0-N
0000573B 6648                   <1>       dec   ax                  ; max. cylinder number
0000573D 88C5                   <1>       MOV   CH,AL
0000573F 66250003               <1>       AND   AX,0300H            ; HIGH TWO BITS OF CYLINDER
00005743 66D1E8                 <1>       SHR   AX,1
00005746 66D1E8                 <1>       SHR   AX,1
                                <1>       ;OR   AL,[ES:BX+14]       ; SECTORS
00005749 0A430E                 <1>       or    al, [ebx+14]
0000574C 88C1                   <1>       MOV   CL,AL
                                <1>       ;MOV  DH,[ES:BX+2]        ; HEADS
0000574E 8A7302                 <1>       mov   dh, [ebx+2]
00005751 FECE                   <1>       DEC   DH                  ; 0-N RANGE
00005753 8A15[B48A0100]         <1>       MOV   DL,[HF_NUM]         ; DRIVE COUNT
00005759 6629C0                 <1>       SUB   AX,AX
                                <1>         ;27/12/2014
                                <1>       ;mov  di, bx              ; HDPT offset
                                <1>
                                <1>       ; 29/08/2020
0000575C 833C2400               <1>       cmp   dword [esp], 0
00005760 7703                   <1>       ja    short G7 ; ebx > 0
                                <1>
                                <1>       ; if EBX (user's buffer address) = 0, do not copy DPT
00005762 5B                     <1>       pop   ebx
00005763 EB24                   <1>       jmp   short G5
                                <1> G7:
                                <1>       ; 27/05/2016
                                <1>       ; return fixed disk parameters table to user
                                <1>       ; in user's buffer, which is pointed by EBX
                                <1>       ;
00005765 873C24                 <1>       xchg  edi, [esp]          ; ebx (input)-> edi, edi -> [esp]
00005768 56                     <1>       push  esi
00005769 89DE                   <1>       mov   esi, ebx            ; hard disk parameter table (32 bytes)
0000576B 89FB                   <1>       mov   ebx, edi            ; ebx = user's buffer address
0000576D 51                     <1>       push  ecx
0000576E 50                     <1>       push  eax
0000576F B920000000             <1>       mov   ecx, 32 ; 32 bytes
00005774 E8CEC30000             <1>       call  transfer_to_user_buffer ; trdosk6.s (16/05/2016)
00005779 58                     <1>       pop   eax
0000577A 59                     <1>       pop   ecx
0000577B 5E                     <1>       pop   esi
```

```
4672 0000577C 5F                <1>       pop    edi
4673 0000577D 730A              <1>       jnc    short G5
4674                            <1>       ; 29/05/2016 (*)
4675 0000577F B8FF000000        <1>       mov    eax, 0FFh ; unknown error !
4676                            <1> G6:
4677 00005784 804C241001        <1>       or     byte [esp+16], 1 ; set carry bit of eflags register
4678                            <1> G5:
4679                            <1>       ; 27/05/2016
4680                            <1>       ;POP   eBX                ; RESTORE REGISTERS
4681 00005789 07                <1>       POP    ES
4682 0000578A 1F                <1>       POP    DS
4683                            <1>       ;RETf  2
4684                            <1>       ; (*) 29/05/2016
4685                            <1>       ; (*) retf 4
4686                            <1>       ; (*) or byte [esp+8], 1 ; set carry bit of eflags register
4687 0000578B CF                <1>       iretd
4688                            <1> G4:
4689 0000578C C605[B38A0100]07  <1>       MOV     byte [DISK_STATUS1],INIT_FAIL ; OPERATION FAILED
4690                            <1>       ;mov   ah, NOT_DRY ; 30/08/2020 - 'drive not ready' error code
4691 00005793 B407              <1>       MOV    AH,INIT_FAIL
4692 00005795 28C0              <1>       SUB    AL,AL
4693                            <1>       ;SUB   DX,DX
4694                            <1>       ; 30/08/2020
4695 00005797 8A15[B48A0100]    <1>       mov    dl, [HF_NUM] ; disk count
4696 0000579D 28F6              <1>       sub    dh, dh
4697 0000579F 6629C9            <1>       SUB    CX,CX
4698                            <1>       ; 29/05/2016 (*)
4699                            <1>       ;STC                      ; SET ERROR FLAG
4700                            <1>       ;JMP   short G5
4701                            <1>       ; 29/08/2020 - BugFix
4702 000057A2 5B                <1>       pop    ebx
4703 000057A3 EBDF              <1>       jmp    short G6
4704                            <1>
4705                            <1> ;--------------------------------------
4706                            <1> ;     INITIALIZE DRIVE     (AH = 09H) :
4707                            <1> ;--------------------------------------
4708                            <1>       ; 03/01/2015
4709                            <1>       ; According to ATA-ATAPI specification v2.0 to v5.0
4710                            <1>       ; logical sector per logical track
4711                            <1>       ; and logical heads - 1 would be set but
4712                            <1>       ; it is seen as it will be good
4713                            <1>       ; if physical parameters will be set here
4714                            <1>       ; because, number of heads <= 16.
4715                            <1>       ; (logical heads usually more than 16)
4716                            <1>       ; NOTE: ATA logical parameters (software C, H, S)
4717                            <1>       ;     == INT 13h physical parameters
4718                            <1>
4719                            <1> ;INIT_DRV:
4720                            <1> ;     MOV    byte [CMD_BLOCK+6],SET_PARM_CMD
4721                            <1> ;     CALL   GET_VEC             ; ES:BX -> PARAMETER BLOCK
4722                            <1> ;     MOV    AL,[ES:BX+2]        ; GET NUMBER OF HEADS
4723                            <1> ;     DEC    AL                  ; CONVERT TO 0-INDEX
4724                            <1> ;     MOV    AH,[CMD_BLOCK+5]    ; GET SDH REGISTER
4725                            <1> ;     AND    AH,0F0H             ; CHANGE HEAD NUMBER
4726                            <1> ;     OR     AH,AL               ; TO MAX HEAD
4727                            <1> ;     MOV    [CMD_BLOCK+5],AH
4728                            <1> ;     MOV    AL,[ES:BX+14]       ; MAX SECTOR NUMBER
4729                            <1> ;     MOV    [CMD_BLOCK+1],AL
4730                            <1> ;     SUB    AX,AX
4731                            <1> ;     MOV    [CMD_BLOCK+3],AL    ; ZERO FLAGS
4732                            <1> ;     CALL   COMMAND             ; TELL CONTROLLER
4733                            <1> ;     JNZ    short INIT_EXIT           ; CONTROLLER BUSY ERROR
4734                            <1> ;     CALL   NOT_BUSY            ; WAIT FOR IT TO BE DONE
4735                            <1> ;     JNZ    short INIT_EXIT           ; TIME OUT
4736                            <1> ;     CALL   CHECK_STATUS
4737                            <1> ;INIT_EXIT:
4738                            <1> ;     RETn
4739                            <1>
4740                            <1> ; 04/01/2015
4741                            <1> ; 02/01/2015 - Derived from from AWARD BIOS 1999
4742                            <1> ;                          AHDSK.ASM - INIT_DRIVE
4743                            <1> INIT_DRV:
4744                            <1>       ;xor   ah,ah
4745 000057A5 31C0              <1>       xor    eax, eax ; 21/02/2015
4746 000057A7 B00B              <1>       mov    al,11 ; Physical heads from translated HDPT
4747 000057A9 3825[C88A0100]    <1>         cmp     [LBAMode], ah   ; 0
4748 000057AF 7702              <1>       ja     short idrv0
4749 000057B1 B002              <1>       mov    al,2  ; Physical heads from standard HDPT
4750                            <1> idrv0:
4751                            <1>       ; DL = drive number (0 based)
4752 000057B3 E8C6030000        <1>       call   GET_VEC
4753                            <1>       ;push  bx
4754 000057B8 53                <1>       push   ebx ; 21/02/2015
4755                            <1>       ;add   bx,ax
4756 000057B9 01C3              <1>       add    ebx, eax
4757                            <1>       ;; 05/01/2015
4758 000057BB 8A25[046E0000]    <1>       mov    ah, [hf_m_s] ; drive number (0= master, 1= slave)
4759                            <1>       ;;and  ah,1
4760 000057C1 C0E404            <1>       shl    ah,4
4761 000057C4 80CCA0            <1>       or     ah,0A0h  ; Drive/Head register - 10100000b (A0h)
4762                            <1>       ;mov   al,[es:bx]
4763 000057C7 8A03              <1>       mov    al, [ebx] ; 21/02/2015
4764 000057C9 FEC8              <1>       dec    al      ; last head number
4765                            <1>       ;and   al,0Fh
4766 000057CB 08E0              <1>       or     al,ah   ; lower 4 bits for head number
4767                            <1>       ;
4768 000057CD C645FE91          <1>       mov    byte [CMD_BLOCK+6],SET_PARM_CMD
4769 000057D1 8845FD            <1>       mov    [CMD_BLOCK+5],al
4770                            <1>       ;pop   bx
4771 000057D4 5B                <1>       pop    ebx
4772 000057D5 29C0              <1>       sub    eax, eax ; 21/02/2015
4773 000057D7 B004              <1>       mov    al,4 ; Physical sec per track from translated HDPT
4774 000057D9 803D[C88A0100]00  <1>       cmp    byte [LBAMode], 0
4775 000057E0 7702              <1>       ja     short idrv1
4776 000057E2 B00E              <1>       mov    al,14 ; Physical sec per track from standard HDPT
```

```
                        <1> idrv1:
                        <1>       ;xor   ah,ah
                        <1>       ;add   bx,ax
000057E4 01C3           <1>       add    ebx, eax ; 21/02/2015
                        <1>       ;mov   al,[es:bx]
                        <1>                      ; sector number
000057E6 8A03           <1>       mov    al, [ebx]
000057E8 8845F9         <1>       mov    [CMD_BLOCK+1],al
000057EB 28C0           <1>       sub    al,al
000057ED 8845FB         <1>       mov    [CMD_BLOCK+3],al  ; ZERO FLAGS
000057F0 E8CA010000     <1>       call   COMMAND         ; TELL CONTROLLER
000057F5 750C           <1>       jnz    short INIT_EXIT       ; CONTROLLER BUSY ERROR
000057F7 E878020000     <1>       call   NOT_BUSY        ; WAIT FOR IT TO BE DONE
000057FC 7505           <1>       jnz    short INIT_EXIT       ; TIME OUT
000057FE E8C9020000     <1>       call   CHECK_STATUS
                        <1> INIT_EXIT:
00005803 C3             <1>       RETn
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     READ LONG         (AH = 0AH) :
                        <1> ;---------------------------------------
                        <1>
                        <1> RD_LONG:
                        <1>       ;MOV   @CMD_BLOCK+6,READ_CMD OR ECC_MODE
00005804 C645FE22       <1>         mov     byte [CMD_BLOCK+6],READ_CMD + ECC_MODE
00005808 E9E0000000     <1>         JMP     COMMANDI
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     WRITE LONG        (AH = 0BH) :
                        <1> ;---------------------------------------
                        <1>
                        <1> WR_LONG:
                        <1>       ;MOV   @CMD_BLOCK+6,WRITE_CMD OR ECC_MODE
0000580D C645FE32       <1>         MOV     byte [CMD_BLOCK+6],WRITE_CMD + ECC_MODE
00005811 E932010000     <1>         JMP     COMMANDO
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     SEEK              (AH = 0CH) :
                        <1> ;---------------------------------------
                        <1>
                        <1> DISK_SEEK:
00005816 C645FE70       <1>         MOV     byte [CMD_BLOCK+6],SEEK_CMD
0000581A E8A0010000     <1>       CALL   COMMAND
0000581F 751C           <1>       JNZ    short DS_EXIT             ; CONTROLLER BUSY ERROR
00005821 E812020000     <1>       CALL   _WAIT
00005826 7515           <1>         JNZ     DS_EXIT                 ; TIME OUT ON SEEK
00005828 E89F020000     <1>       CALL   CHECK_STATUS
0000582D 803D[B38A0100]40 <1>         CMP     byte [DISK_STATUS1],BAD_SEEK
00005834 7507           <1>       JNE    short DS_EXIT
00005836 C605[B38A0100]00 <1>         MOV     byte [DISK_STATUS1],0
                        <1> DS_EXIT:
0000583D C3             <1>       RETn
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     TEST DISK READY      (AH = 10H) :
                        <1> ;---------------------------------------
                        <1>
                        <1> TST_RDY:                          ; WAIT FOR CONTROLLER
0000583E E831020000     <1>       CALL   NOT_BUSY
00005843 751C           <1>       JNZ    short TR_EX
00005845 8A45FD         <1>       MOV    AL,[CMD_BLOCK+5]    ; SELECT DRIVE
00005848 668B15[006E0000] <1>     MOV    DX,[HF_PORT]
0000584F 80C206         <1>       add    dl,6
00005852 EE             <1>       OUT    DX,AL
00005853 E88C020000     <1>       CALL   CHECK_ST            ; CHECK STATUS ONLY
00005858 7507           <1>       JNZ    short TR_EX
0000585A C605[B38A0100]00 <1>     MOV    byte [DISK_STATUS1],0     ; WIPE OUT DATA CORRECTED ERROR
                        <1> TR_EX:
00005861 C3             <1>       RETn
                        <1>
                        <1> ;---------------------------------------
                        <1> ;     RECALIBRATE       (AH = 11H) :
                        <1> ;---------------------------------------
                        <1>
                        <1> HDISK_RECAL:
00005862 C645FE10       <1>         MOV     byte [CMD_BLOCK+6],RECAL_CMD ; 10h, 16
00005866 E854010000     <1>       CALL   COMMAND             ; START THE OPERATION
0000586B 7523           <1>       JNZ    short RECAL_EXIT    ; ERROR
0000586D E8C6010000     <1>       CALL   _WAIT               ; WAIT FOR COMPLETION
00005872 7407           <1>       JZ     short RECAL_X             ; TIME OUT ONE OK ?
00005874 E8BF010000     <1>       CALL   _WAIT               ; WAIT FOR COMPLETION LONGER
00005879 7515           <1>       JNZ    short RECAL_EXIT    ; TIME OUT TWO TIMES IS ERROR
                        <1> RECAL_X:
0000587B E84C020000     <1>       CALL   CHECK_STATUS
00005880 803D[B38A0100]40 <1>     CMP    byte [DISK_STATUS1],BAD_SEEK ; SEEK NOT COMPLETE
00005887 7507           <1>       JNE    short RECAL_EXIT    ; IS OK
00005889 C605[B38A0100]00 <1>     MOV    byte [DISK_STATUS1],0
                        <1> RECAL_EXIT:
00005890 803D[B38A0100]00 <1>         CMP     byte [DISK_STATUS1],0
00005897 C3             <1>       RETn
                        <1>
                        <1> ;---------------------------------------
                        <1> ;      CONTROLLER DIAGNOSTIC (AH = 14H) :
                        <1> ;---------------------------------------
                        <1>
                        <1> CTLR_DIAGNOSTIC:
00005898 FA             <1>         CLI                             ; DISABLE INTERRUPTS WHILE CHANGING MASK
00005899 E4A1           <1>       IN     AL,INTB01          ; TURN ON SECOND INTERRUPT CHIP
                        <1>       ;AND   AL,0BFH
0000589B 243F           <1>       and    al, 3Fh                  ; enable IRQ 14 & IRQ 15
                        <1>       ;JMP   $+2
                        <1>       IODELAY
0000589D EB00           <2>  jmp short $+2
0000589F EB00           <2>  jmp short $+2
000058A1 E6A1           <1>       OUT    INTB01,AL
```

```
                                <1>       IODELAY
000058A3 EB00                   <2>  jmp short $+2
000058A5 EB00                   <2>  jmp short $+2
000058A7 E421                   <1>       IN     AL,INTA01          ; LET INTERRUPTS PASS THRU TO
000058A9 24FB                   <1>       AND    AL,0FBH            ;  SECOND CHIP
                                <1>       ;JMP   $+2
                                <1>       IODELAY
000058AB EB00                   <2>  jmp short $+2
000058AD EB00                   <2>  jmp short $+2
000058AF E621                   <1>       OUT    INTA01,AL
000058B1 FB                     <1>       STI
000058B2 E8BD010000             <1>       CALL   NOT_BUSY           ; WAIT FOR CARD
000058B7 752B                   <1>       JNZ    short CD_ERR       ; BAD CARD
                                <1>       ;MOV   DX, HF_PORT+7
000058B9 668B15[006E0000]       <1>       mov    dx, [HF_PORT]
000058C0 80C207                 <1>       add    dl, 7
000058C3 B090                   <1>       MOV    AL,DIAG_CMD        ; START DIAGNOSE
000058C5 EE                     <1>       OUT    DX,AL
000058C6 E8A9010000             <1>       CALL   NOT_BUSY           ; WAIT FOR IT TO COMPLETE
000058CB B480                   <1>       MOV    AH,TIME_OUT
000058CD 7517                   <1>       JNZ    short CD_EXIT            ; TIME OUT ON DIAGNOSTIC
                                <1>       ;MOV   DX,HF_PORT+1       ; GET ERROR REGISTER
000058CF 668B15[006E0000]       <1>       mov    dx, [HF_PORT]
000058D6 FEC2                   <1>       inc    dl
000058D8 EC                     <1>       IN     AL,DX
000058D9 A2[AA8A0100]           <1>       MOV    [HF_ERROR],AL      ; SAVE IT
000058DE B400                   <1>       MOV    AH,0
000058E0 3C01                   <1>       CMP    AL,1               ; CHECK FOR ALL OK
000058E2 7402                   <1>       JE     SHORT CD_EXIT
000058E4 B420                   <1> CD_ERR: MOV  AH,BAD_CNTLR
                                <1> CD_EXIT:
000058E6 8825[B38A0100]         <1>       MOV    [DISK_STATUS1],AH
000058EC C3                     <1>       RETn
                                <1>
                                <1> ;---------------------------------------
                                <1> ; COMMANDI                        :
                                <1> ;     REPEATEDLY INPUTS DATA TILL     :
                                <1> ;     NSECTOR RETURNS ZERO            :
                                <1> ;---------------------------------------
                                <1> COMMANDI:
000058ED E862020000             <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
000058F2 7253                   <1>       JC     short CMD_ABORT
                                <1>       ;MOV   DI,BX
000058F4 89DF                   <1>       mov    edi, ebx ; 21/02/2015
000058F6 E8C4000000             <1>       CALL   COMMAND            ; OUTPUT COMMAND
000058FB 754A                   <1>       JNZ    short CMD_ABORT
                                <1> CMD_I1:
000058FD E836010000             <1>       CALL   _WAIT              ; WAIT FOR DATA REQUEST INTERRUPT
00005902 7543                   <1>       JNZ    short TM_OUT       ; TIME OUT
                                <1> cmd_i1x: ; 18/02/2016
                                <1>       ;MOV   CX,256             ; SECTOR SIZE IN WORDS
00005904 B900010000             <1>       mov    ecx, 256 ; 21/02/2015
                                <1>       ;MOV   DX,HF_PORT
00005909 668B15[006E0000]       <1>       mov    dx,[HF_PORT]
00005910 FA                     <1>       CLI
00005911 FC                     <1>       CLD
00005912 F3666D                 <1>       REP    INSW               ; GET THE SECTOR
00005915 FB                     <1>       STI
00005916 F645FE02               <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL INPUT
0000591A 7419                   <1>       JZ     short CMD_I3
0000591C E880010000             <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
00005921 7224                   <1>       JC     short TM_OUT
                                <1>       ;MOV   DX,HF_PORT
00005923 668B15[006E0000]       <1>       mov    dx,[HF_PORT]
                                <1>       ;MOV   CX,4               ; GET ECC BYTES
0000592A B904000000             <1>       mov    ecx, 4 ; mov cx, 4
0000592F EC                     <1> CMD_I2: IN   AL,DX
                                <1>       ;MOV   [ES:DI],AL         ; GO SLOW FOR BOARD
00005930 8807                   <1>       mov    [edi], al ; 21/02/2015
00005932 47                     <1>       INC    eDI
00005933 E2FA                   <1>       LOOP   CMD_I2
                                <1> CMD_I3:
                                <1>       ; wait for 400 ns
00005935 80C207                 <1>       add    dl, 7
00005938 EC                     <1>       in     al, dx
00005939 EC                     <1>       in     al, dx
0000593A EC                     <1>       in     al, dx
                                <1>       ;
0000593B E88C010000             <1>       CALL   CHECK_STATUS
00005940 7505                   <1>       JNZ    short CMD_ABORT          ; ERROR RETURNED
00005942 FE4DF9                 <1>       DEC    byte [CMD_BLOCK+1]  ; CHECK FOR MORE
                                <1>       ;JNZ   SHORT CMD_I1
00005945 75BD                   <1>       jnz    short cmd_i1x ; 18/02/2016
                                <1> CMD_ABORT:
00005947 C3                     <1> TM_OUT: RETn
                                <1>
                                <1> ;---------------------------------------
                                <1> ; COMMANDO                        :
                                <1> ;     REPEATEDLY OUTPUTS DATA TILL    :
                                <1> ;     NSECTOR RETURNS ZERO            :
                                <1> ;---------------------------------------
                                <1> COMMANDO:
00005948 E807020000             <1>       CALL   CHECK_DMA          ; CHECK 64K BOUNDARY ERROR
0000594D 72F8                   <1>       JC     short CMD_ABORT
0000594F 89DE                   <1> CMD_OF: MOV  eSI,eBX ; 21/02/2015
00005951 E869000000             <1>       CALL   COMMAND            ; OUTPUT COMMAND
00005956 75EF                   <1>       JNZ    short CMD_ABORT
00005958 E844010000             <1>       CALL   WAIT_DRQ           ; WAIT FOR DATA REQUEST
0000595D 72E8                   <1>       JC     short TM_OUT              ; TOO LONG
                                <1> CMD_O1: ;PUSH      DS
                                <1>       ;PUSH  ES                 ; MOVE ES TO DS
                                <1>       ;POP   DS
                                <1>       ;MOV   CX,256             ; PUT THE DATA OUT TO THE CARD
                                <1>       ;MOV   DX,HF_PORT
                                <1>       ; 01/02/2015
```

```
4981 0000595F 668B15[006E0000]    <1>       mov    dx, [HF_PORT]
4982                              <1>       ;push  es
4983                              <1>       ;pop   ds
4984                              <1>       ;mov   cx, 256
4985 00005966 B900010000          <1>       mov    ecx, 256 ; 21/02/2015
4986 0000596B FA                  <1>       CLI
4987 0000596C FC                  <1>       CLD
4988 0000596D F3666F              <1>       REP    OUTSW
4989 00005970 FB                  <1>       STI
4990                              <1>       ;POP   DS                  ; RESTORE DS
4991 00005971 F645FE02            <1>       TEST   byte [CMD_BLOCK+6],ECC_MODE ; CHECK FOR NORMAL OUTPUT
4992 00005975 7419                <1>       JZ     short CMD_O3
4993 00005977 E825010000          <1>       CALL   WAIT_DRQ            ; WAIT FOR DATA REQUEST
4994 0000597C 72C9                <1>       JC     short TM_OUT
4995                              <1>       ;MOV   DX,HF_PORT
4996 0000597E 668B15[006E0000]    <1>       mov    dx, [HF_PORT]
4997                              <1>       ;MOV   CX,4                ; OUTPUT THE ECC BYTES
4998 00005985 B904000000          <1>       mov    ecx, 4  ; mov cx, 4
4999                              <1> CMD_O2: ;MOV AL,[ES:SI]
5000 0000598A 8A06                <1>       mov    al, [esi]
5001 0000598C EE                  <1>       OUT    DX,AL
5002 0000598D 46                  <1>       INC    eSI
5003 0000598E E2FA                <1>       LOOP   CMD_O2
5004                              <1> CMD_O3:
5005 00005990 E8A3000000          <1>       CALL   _WAIT               ; WAIT FOR SECTOR COMPLETE INTERRUPT
5006 00005995 75B0                <1>       JNZ    short TM_OUT        ; ERROR RETURNED
5007 00005997 E830010000          <1>       CALL   CHECK_STATUS
5008 0000599C 75A9                <1>       JNZ    short CMD_ABORT
5009 0000599E F605[A98A0100]08    <1>       TEST   byte [HF_STATUS],ST_DRQ   ; CHECK FOR MORE
5010 000059A5 75B8                <1>       JNZ    SHORT CMD_O1
5011                              <1>       ;MOV   DX,HF_PORT+2        ; CHECK RESIDUAL SECTOR COUNT
5012 000059A7 668B15[006E0000]    <1>       mov    dx, [HF_PORT]
5013                              <1>       ;add   dl, 2
5014 000059AE FEC2                <1>       inc    dl
5015 000059B0 FEC2                <1>       inc    dl
5016 000059B2 EC                  <1>       IN     AL,DX               ;
5017 000059B3 A8FF                <1>       TEST   AL,0FFH             ;
5018 000059B5 7407                <1>       JZ     short CMD_O4              ; COUNT = 0  OK
5019 000059B7 C605[B38A0100]BB    <1>       MOV    byte [DISK_STATUS1],UNDEF_ERR
5020                              <1>                                  ; OPERATION ABORTED - PARTIAL TRANSFER
5021                              <1> CMD_O4:
5022 000059BE C3                  <1>       RETn
5023                              <1>
5024                              <1> ;--------------------------------------------------------
5025                              <1> ; COMMAND                                          :
5026                              <1> ;     THIS ROUTINE OUTPUTS THE COMMAND BLOCK       :
5027                              <1> ; OUTPUT                                           :
5028                              <1> ;     BL = STATUS                                  :
5029                              <1> ;     BH = ERROR REGISTER                          :
5030                              <1> ;--------------------------------------------------------
5031                              <1>
5032                              <1> COMMAND:
5033 000059BF 53                  <1>       PUSH   eBX                 ; WAIT FOR SEEK COMPLETE AND READY
5034                              <1>       ;;MOV  CX,DELAY_2          ; SET INITIAL DELAY BEFORE TEST
5035                              <1> COMMAND1:
5036                              <1>       ;;PUSH CX                  ; SAVE LOOP COUNT
5037 000059C0 E879FEFFFF          <1>       CALL   TST_RDY             ; CHECK DRIVE READY
5038                              <1>       ;;POP  CX
5039 000059C5 7419                <1>       JZ     short COMMAND2             ; DRIVE IS READY
5040 000059C7 803D[B38A0100]80    <1>         CMP     byte [DISK_STATUS1],TIME_OUT ; TST_RDY TIMED OUT--GIVE UP
5041                              <1>       ;JZ    short CMD_TIMEOUT
5042                              <1>       ;;LOOP COMMAND1            ; KEEP TRYING FOR A WHILE
5043                              <1>       ;JMP   SHORT COMMAND4             ; ITS NOT GOING TO GET READY
5044 000059CE 7507                <1>       jne    short COMMAND4
5045                              <1> CMD_TIMEOUT:
5046 000059D0 C605[B38A0100]20    <1>       MOV    byte [DISK_STATUS1],BAD_CNTLR
5047                              <1> COMMAND4:
5048 000059D7 5B                  <1>       POP    eBX
5049 000059D8 803D[B38A0100]00    <1>         CMP     byte [DISK_STATUS1],0   ; SET CONDITION CODE FOR CALLER
5050 000059DF C3                  <1>       RETn
5051                              <1> COMMAND2:
5052 000059E0 5B                  <1>       POP    eBX
5053 000059E1 57                  <1>       PUSH   eDI
5054 000059E2 C605[AB8A0100]00    <1>       MOV    byte [HF_INT_FLAG],0      ; RESET INTERRUPT FLAG
5055 000059E9 FA                  <1>       CLI                        ; INHIBIT INTERRUPTS WHILE CHANGING MASK
5056 000059EA E4A1                <1>       IN     AL,INTB01           ; TURN ON SECOND INTERRUPT CHIP
5057                              <1>       ;AND   AL,0BFH
5058 000059EC 243F                <1>       and    al, 3Fh                   ; Enable IRQ 14 & 15
5059                              <1>       ;JMP   $+2
5060                              <1>       IODELAY
5060 000059EE EB00                <2>  jmp short $+2
5060 000059F0 EB00                <2>  jmp short $+2
5061 000059F2 E6A1                <1>       OUT    INTB01,AL
5062 000059F4 E421                <1>       IN     AL,INTA01           ; LET INTERRUPTS PASS THRU TO
5063 000059F6 24FB                <1>       AND    AL,0FBH             ;  SECOND CHIP
5064                              <1>       ;JMP   $+2
5065                              <1>       IODELAY
5065 000059F8 EB00                <2>  jmp short $+2
5065 000059FA EB00                <2>  jmp short $+2
5066 000059FC E621                <1>       OUT    INTA01,AL
5067 000059FE FB                  <1>       STI
5068 000059FF 31FF                <1>       XOR    eDI,eDI                   ; INDEX THE COMMAND TABLE
5069                              <1>       ;MOV   DX,HF_PORT+1        ; DISK ADDRESS
5070 00005A01 668B15[006E0000]    <1>       mov    dx, [HF_PORT]
5071 00005A08 FEC2                <1>       inc    dl
5072 00005A0A F605[B58A0100]C0    <1>       TEST   byte [CONTROL_BYTE],0C0H ; CHECK FOR RETRY SUPPRESSION
5073 00005A11 7411                <1>       JZ     short COMMAND3
5074 00005A13 8A45FE              <1>       MOV    AL, [CMD_BLOCK+6]   ; YES-GET OPERATION CODE
5075 00005A16 24F0                <1>       AND    AL,0F0H             ; GET RID OF MODIFIERS
5076 00005A18 3C20                <1>       CMP    AL,20H              ; 20H-40H IS READ, WRITE, VERIFY
5077 00005A1A 7208                <1>       JB     short COMMAND3
5078 00005A1C 3C40                <1>       CMP    AL,40H
5079 00005A1E 7704                <1>       JA     short COMMAND3
5080 00005A20 804DFE01            <1>       OR     byte [CMD_BLOCK+6],NO_RETRIES
5081                              <1>                                  ; VALID OPERATION FOR RETRY SUPPRESS
```

```
                            <1> COMMAND3:
00005A24 8A443DF8           <1>       MOV    AL,[CMD_BLOCK+eDI]  ; GET THE COMMAND STRING BYTE
00005A28 EE                 <1>       OUT    DX,AL               ; GIVE IT TO CONTROLLER
                            <1>       IODELAY
00005A29 EB00               <2>  jmp short $+2
00005A2B EB00               <2>  jmp short $+2
00005A2D 47                 <1>       INC    eDI                 ; NEXT BYTE IN COMMAND BLOCK
00005A2E 6642               <1>       INC    DX                  ; NEXT DISK ADAPTER REGISTER
00005A30 6683FF07           <1>       cmp    di, 7  ; 1/1/2015   ; ALL DONE?
00005A34 75EE               <1>       JNZ    short COMMAND3            ; NO--GO DO NEXT ONE
00005A36 5F                 <1>       POP    eDI
00005A37 C3                 <1>       RETn                       ; ZERO FLAG IS SET
                            <1>
                            <1> ;CMD_TIMEOUT:
                            <1> ;     MOV    byte [DISK_STATUS1],BAD_CNTLR
                            <1> ;COMMAND4:
                            <1> ;     POP    BX
                            <1> ;     CMP    [DISK_STATUS1],0    ; SET CONDITION CODE FOR CALLER
                            <1> ;     RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     WAIT FOR INTERRUPT        :
                            <1> ;----------------------------------------
                            <1> ;WAIT:
                            <1> _WAIT:
00005A38 FB                 <1>       STI                        ; MAKE SURE INTERRUPTS ARE ON
                            <1>       ;SUB   CX,CX               ; SET INITIAL DELAY BEFORE TEST
                            <1>       ;CLC
                            <1>       ;MOV   AX,9000H            ; DEVICE WAIT INTERRUPT
                            <1>       ;INT   15H
                            <1>       ;JC    WT2                 ; DEVICE TIMED OUT
                            <1>       ;MOV   BL,DELAY_1          ; SET DELAY COUNT
                            <1>
                            <1>       ;mov   bl, WAIT_HDU_INT_HI
                            <1>       ;; 21/02/2015
                            <1>       ;;mov  bl, WAIT_HDU_INT_HI + 1
                            <1>       ;;mov  cx, WAIT_HDU_INT_LO
00005A39 B915160500         <1>       mov    ecx, WAIT_HDU_INT_LH
                            <1>                                  ; (AWARD BIOS -> WAIT_FOR_MEM)
                            <1> ;-----      WAIT LOOP
                            <1>
                            <1> WT1:
                            <1>       ;TEST  byte [HF_INT_FLAG],80H    ; TEST FOR INTERRUPT
00005A3E F605[AB8A0100]C0   <1>       test   byte [HF_INT_FLAG],0C0h
                            <1>       ;LOOPZ WT1
00005A45 7517               <1>       JNZ    short WT3           ; INTERRUPT--LETS GO
                            <1>       ;DEC   BL
                            <1>       ;JNZ   short WT1           ; KEEP TRYING FOR A WHILE
                            <1>
                            <1> WT1_hi:
00005A47 E461               <1>       in     al, SYS1 ; 61h (PORT_B)   ; wait for lo to hi
00005A49 A810               <1>       test   al, 10h                   ; transition on memory
00005A4B 75FA               <1>       jnz    short WT1_hi        ; refresh.
                            <1> WT1_lo:
00005A4D E461               <1>       in     al, SYS1            ; 061h (PORT_B)
00005A4F A810               <1>       test   al, 10h
00005A51 74FA               <1>       jz     short WT1_lo
00005A53 E2E9               <1>       loop   WT1
                            <1>       ;;or   bl, bl
                            <1>       ;;jz   short WT2
                            <1>       ;;dec  bl
                            <1>       ;;jmp  short WT1
                            <1>       ;dec   bl
                            <1>       ;jnz   short WT1
                            <1>
00005A55 C605[B38A0100]80   <1> WT2:  MOV    byte [DISK_STATUS1],TIME_OUT ; REPORT TIME OUT ERROR
00005A5C EB0E               <1>       JMP    SHORT WT4
00005A5E C605[B38A0100]00   <1> WT3:  MOV    byte [DISK_STATUS1],0
00005A65 C605[AB8A0100]00   <1>       MOV    byte [HF_INT_FLAG],0
00005A6C 803D[B38A0100]00   <1> WT4:  CMP    byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
00005A73 C3                 <1>       RETn
                            <1>
                            <1> ;----------------------------------------
                            <1> ;     WAIT FOR CONTROLLER NOT BUSY     :
                            <1> ;----------------------------------------
                            <1> NOT_BUSY:
00005A74 FB                 <1>       STI                        ; MAKE SURE INTERRUPTS ARE ON
                            <1>       ;PUSH  eBX
                            <1>       ;SUB   CX,CX               ; SET INITIAL DELAY BEFORE TEST
00005A75 668B15[006E0000]   <1>       mov    DX, [HF_PORT]
00005A7C 80C207             <1>       add    dl, 7               ; Status port (HF_PORT+7)
                            <1>       ;MOV   BL,DELAY_1
                            <1>                                  ; wait for 10 seconds
                            <1>       ;mov   cx, WAIT_HDU_INT_LO ; 1615h
                            <1>       ;;mov  bl, WAIT_HDU_INT_HI ;   05h
                            <1>       ;mov   bl, WAIT_HDU_INT_HI + 1
00005A7F B915160500         <1>       mov    ecx, WAIT_HDU_INT_LH  ; 21/02/2015
                            <1>       ;
                            <1> ;;      mov     byte [wait_count], 0    ; Reset wait counter
                            <1> NB1:
00005A84 EC                 <1>       IN     AL,DX               ; CHECK STATUS
                            <1>       ;TEST  AL,ST_BUSY
00005A85 2480               <1>       and    al, ST_BUSY
                            <1>       ;LOOPNZ      NB1
00005A87 7410               <1>       JZ     short NB2           ; NOT BUSY--LETS GO
                            <1>       ;DEC   BL
                            <1>       ;JNZ   short NB1           ; KEEP TRYING FOR A WHILE
                            <1>
00005A89 E461               <1> NB1_hi: IN   AL,SYS1                   ; wait for hi to lo
00005A8B A810               <1>       TEST   AL,010H                   ; transition on memory
00005A8D 75FA               <1>       JNZ    SHORT NB1_hi        ; refresh.
00005A8F E461               <1> NB1_lo: IN   AL,SYS1
00005A91 A810               <1>       TEST   AL,010H
00005A93 74FA               <1>       JZ     short NB1_lo
00005A95 E2ED               <1>       LOOP   NB1
```

```
5185                                <1>       ;dec   bl
5186                                <1>       ;jnz   short NB1
5187                                <1>       ;
5188                                <1> ;;      cmp     byte [wait_count], 182  ; 10 seconds (182 timer ticks)
5189                                <1> ;;    jb     short NB1
5190                                <1>       ;
5191                                <1>       ;MOV   [DISK_STATUS1],TIME_OUT   ; REPORT TIME OUT ERROR
5192                                <1>       ;JMP   SHORT NB3
5193 00005A97 B080                 <1>       mov    al, TIME_OUT
5194                                <1> NB2:
5195                                <1>       ;MOV   byte [DISK_STATUS1],0
5196                                <1> ;NB3:
5197                                <1>       ;POP   eBX
5198 00005A99 A2[B38A0100]         <1>       mov    [DISK_STATUS1], al  ;;; will be set after return
5199                                <1>       ;CMP   byte [DISK_STATUS1],0     ; SET CONDITION CODE FOR CALLER
5200 00005A9E 08C0                 <1>       or     al, al              ; (zf = 0 --> timeout)
5201 00005AA0 C3                   <1>       RETn
5202                                <1>
5203                                <1> ;----------------------------------------
5204                                <1> ;     WAIT FOR DATA REQUEST            :
5205                                <1> ;----------------------------------------
5206                                <1> WAIT_DRQ:
5207                                <1>       ;MOV   CX,DELAY_3
5208                                <1>       ;MOV   DX,HF_PORT+7
5209 00005AA1 668B15[006E0000]     <1>       mov    dx, [HF_PORT]
5210 00005AA8 80C207               <1>       add    dl, 7
5211                                <1>       ;;MOV  bl, WAIT_HDU_DRQ_HI ; 0
5212                                <1>       ;MOV   cx, WAIT_HDU_DRQ_LO ; 1000 (30 milli seconds)
5213                                <1>                                  ; (but it is written as 2000
5214                                <1>                                  ; micro seconds in ATORGS.ASM file
5215                                <1>                                  ; of Award Bios - 1999, D1A0622)
5216 00005AAB B9E8030000           <1>       mov    ecx, WAIT_HDU_DRQ_LH ; 21/02/2015
5217 00005AB0 EC                   <1> WQ_1: IN     AL,DX               ; GET STATUS
5218 00005AB1 A808                 <1>       TEST   AL,ST_DRQ           ; WAIT FOR DRQ
5219 00005AB3 7516                 <1>       JNZ    short WQ_OK
5220                                <1>       ;LOOP  WQ_1                ; KEEP TRYING FOR A SHORT WHILE
5221                                <1> WQ_hi:
5222 00005AB5 E461                 <1>       IN     AL,SYS1                   ; wait for hi to lo
5223 00005AB7 A810                 <1>       TEST   AL,010H                   ; transition on memory
5224 00005AB9 75FA                 <1>       JNZ    SHORT WQ_hi         ; refresh.
5225 00005ABB E461                 <1> WQ_lo:  IN      AL,SYS1
5226 00005ABD A810                 <1>       TEST   AL,010H
5227 00005ABF 74FA                 <1>       JZ     SHORT WQ_lo
5228 00005AC1 E2ED                 <1>       LOOP   WQ_1
5229                                <1>
5230 00005AC3 C605[B38A0100]80     <1>         MOV     byte [DISK_STATUS1],TIME_OUT ; ERROR
5231 00005ACA F9                   <1>       STC
5232                                <1> WQ_OK:
5233 00005ACB C3                   <1>       RETn
5234                                <1> ;WQ_OK:     ;CLC
5235                                <1> ;     RETn
5236                                <1>
5237                                <1> ;----------------------------------------
5238                                <1> ;     CHECK FIXED DISK STATUS   :
5239                                <1> ;----------------------------------------
5240                                <1> CHECK_STATUS:
5241 00005ACC E813000000           <1>       CALL   CHECK_ST            ; CHECK THE STATUS BYTE
5242 00005AD1 7509                 <1>       JNZ    short CHECK_S1            ; AN ERROR WAS FOUND
5243 00005AD3 A801                 <1>       TEST   AL,ST_ERROR         ; WERE THERE ANY OTHER ERRORS
5244 00005AD5 7405                 <1>       JZ     short CHECK_S1            ; NO ERROR REPORTED
5245 00005AD7 E849000000           <1>       CALL   CHECK_ER            ; ERROR REPORTED
5246                                <1> CHECK_S1:
5247 00005ADC 803D[B38A0100]00     <1>       CMP    byte [DISK_STATUS1],0     ; SET STATUS FOR CALLER
5248 00005AE3 C3                   <1>       RETn
5249                                <1>
5250                                <1> ;----------------------------------------
5251                                <1> ;     CHECK FIXED DISK STATUS BYTE     :
5252                                <1> ;----------------------------------------
5253                                <1> CHECK_ST:
5254                                <1>       ;MOV   DX,HF_PORT+7        ; GET THE STATUS
5255 00005AE4 668B15[006E0000]     <1>       mov    dx, [HF_PORT]
5256 00005AEB 80C207               <1>       add    dl, 7
5257                                <1>
5258                                <1>       ; 17/02/2016
5259                                <1>       ;(http://wiki.osdev.org/ATA_PIO_Mode)
5260                                <1>       ;"delay 400ns to allow drive to set new values of BSY and DRQ"
5261 00005AEE EC                   <1>       IN     AL,DX
5262                                <1>       ;in    al, dx ; 100ns
5263                                <1>       ;in    al, dx ; 100ns
5264                                <1>       ;in    al, dx ; 100ns
5265                                <1>       NEWIODELAY ; 18/02/2016 (AWARD BIOS - 1999, 'CKST' in AHSDK.ASM)
5265 00005AEF E6EB                 <2>  out 0ebh,al
5266                                <1>       ;
5267 00005AF1 A2[A98A0100]         <1>       MOV    [HF_STATUS],AL
5268 00005AF6 B400                 <1>       MOV    AH,0
5269 00005AF8 A880                 <1>       TEST   AL,ST_BUSY          ; IF STILL BUSY
5270 00005AFA 751A                 <1>       JNZ    short CKST_EXIT           ;  REPORT OK
5271 00005AFC B4CC                 <1>       MOV    AH,WRITE_FAULT
5272 00005AFE A820                 <1>       TEST   AL,ST_WRT_FLT       ; CHECK FOR WRITE FAULT
5273 00005B00 7514                 <1>       JNZ    short CKST_EXIT
5274 00005B02 B4AA                 <1>       MOV    AH,NOT_RDY
5275 00005B04 A840                 <1>       TEST   AL,ST_READY         ; CHECK FOR NOT READY
5276 00005B06 740E                 <1>       JZ     short CKST_EXIT
5277 00005B08 B440                 <1>       MOV    AH,BAD_SEEK
5278 00005B0A A810                 <1>       TEST   AL,ST_SEEK_COMPL    ; CHECK FOR SEEK NOT COMPLETE
5279 00005B0C 7408                 <1>       JZ     short CKST_EXIT
5280 00005B0E B411                 <1>       MOV    AH,DATA_CORRECTED
5281 00005B10 A804                 <1>       TEST   AL,ST_CORRCTD       ; CHECK FOR CORRECTED ECC
5282 00005B12 7502                 <1>       JNZ    short CKST_EXIT
5283 00005B14 B400                 <1>       MOV    AH,0
5284                                <1> CKST_EXIT:
5285 00005B16 8825[B38A0100]       <1>       MOV    [DISK_STATUS1],AH   ; SET ERROR FLAG
5286 00005B1C 80FC11               <1>       CMP    AH,DATA_CORRECTED   ; KEEP GOING WITH DATA CORRECTED
5287 00005B1F 7403                 <1>       JZ     short CKST_EX1
5288 00005B21 80FC00               <1>       CMP    AH,0
```

```
                                <1> CKST_EX1:
00005B24 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
00005B25 668B15[006E0000]       <1>       mov    dx, [HF_PORT]       ;
00005B2C FEC2                   <1>       inc    dl
00005B2E EC                     <1>       IN     AL,DX
00005B2F A2[AA8A0100]           <1>       MOV    [HF_ERROR],AL
00005B34 53                     <1>       PUSH   eBX ; 21/02/2015
00005B35 B908000000             <1>       MOV    eCX,8               ; TEST ALL 8 BITS
00005B3A D0E0                   <1> CK1:  SHL    AL,1                ; MOVE NEXT ERROR BIT TO CARRY
00005B3C 7202                   <1>       JC     short CK2           ; FOUND THE ERROR
00005B3E E2FA                   <1>       LOOP   CK1                 ; KEEP TRYING
00005B40 BB[F46D0000]           <1> CK2:  MOV    eBX, ERR_TBL        ; COMPUTE ADDRESS OF
00005B45 01CB                   <1>       ADD    eBX,eCX                   ; ERROR CODE
                                <1>       ;;MOV  AH,BYTE [CS:BX]           ; GET ERROR CODE
                                <1>       ;mov   ah, [bx]
00005B47 8A23                   <1>       mov    ah, [ebx] ; 21/02/2015
00005B49 8825[B38A0100]         <1> CKEX: MOV    [DISK_STATUS1],AH   ; SAVE ERROR CODE
00005B4F 5B                     <1>       POP    eBX
00005B50 80FC00                 <1>       CMP    AH,0
00005B53 C3                     <1>       RETn
                                <1>
                                <1> ;--------------------------------------------------------
                                <1> ; CHECK_DMA                                :
                                <1> ;  -CHECK ES:BX AND # SECTORS TO MAKE SURE THAT IT WILL :
                                <1> ;   FIT WITHOUT SEGMENT OVERFLOW.                    :
                                <1> ;  -ES:BX HAS BEEN REVISED TO THE FORMAT SSSS:000X   :
                                <1> ;  -OK IF # SECTORS < 80H (7FH IF LONG READ OR WRITE)     :
                                <1> ;  -OK IF # SECTORS = 80H (7FH) AND BX <= 00H (04H) :
                                <1> ;  -ERROR OTHERWISE                                  :
                                <1> ;--------------------------------------------------------
                                <1> CHECK_DMA:
00005B54 6650                   <1>       PUSH   AX                  ; SAVE REGISTERS
00005B56 66B80080               <1>       MOV    AX,8000H            ; AH = MAX # SECTORS AL = MAX OFFSET
00005B5A F645FE02               <1>         TEST    byte [CMD_BLOCK+6],ECC_MODE
00005B5E 7404                   <1>       JZ     short CKD1
00005B60 66B8047F               <1>       MOV    AX,7F04H            ; ECC IS 4 MORE BYTES
00005B64 3A65F9                 <1> CKD1: CMP    AH, [CMD_BLOCK+1]   ; NUMBER OF SECTORS
00005B67 7706                   <1>       JA     short CKDOK         ; IT WILL FIT
00005B69 7208                   <1>       JB     short CKDERR        ; TOO MANY
00005B6B 38D8                   <1>       CMP    AL,BL               ; CHECK OFFSET ON MAX SECTORS
00005B6D 7204                   <1>       JB     short CKDERR        ; ERROR
00005B6F F8                     <1> CKDOK:       CLC                       ; CLEAR CARRY
00005B70 6658                   <1>       POP    AX
00005B72 C3                     <1>       RETn                       ; NORMAL RETURN
00005B73 F9                     <1> CKDERR: STC                      ; INDICATE ERROR
00005B74 C605[B38A0100]09       <1>         MOV     byte [DISK_STATUS1],DMA_BOUNDARY
00005B7B 6658                   <1>       POP    AX
00005B7D C3                     <1>       RETn
                                <1>
                                <1> ;---------------------------------------
                                <1> ;     SET UP ES:BX-> DISK PARMS :
                                <1> ;---------------------------------------
                                <1>
                                <1> ; INPUT -> DL = 0 based drive number
                                <1> ; OUTPUT -> ES:BX = disk parameter table address
                                <1>
                                <1> GET_VEC:
                                <1>       ;SUB   AX,AX               ; GET DISK PARAMETER ADDRESS
                                <1>       ;MOV   ES,AX
                                <1>       ;TEST  DL,1
                                <1>       ;JZ    short GV_0
                                <1> ;     LES    BX,[HF1_TBL_VEC]    ; ES:BX -> DRIVE PARAMETERS
                                <1> ;     JMP    SHORT GV_EXIT
                                <1> ;GV_0:
                                <1> ;     LES    BX,[HF_TBL_VEC]           ; ES:BX -> DRIVE PARAMETERS
                                <1> ;
                                <1>       ;xor   bh, bh
00005B7E 31DB                   <1>       xor    ebx, ebx
00005B80 88D3                   <1>       mov    bl, dl
                                <1>       ;;02/01/2015
                                <1>       ;;shl  bl, 1              ; port address offset
                                <1>       ;;mov  ax, [bx+hd_ports]  ; Base port address (1F0h, 170h)
                                <1>       ;;shl  bl, 1              ; dpt pointer offset
00005B82 C0E302                 <1>       shl    bl, 2  ;;
                                <1>       ;add   bx, HF_TBL_VEC            ; Disk parameter table pointer
00005B85 81C3[B88A0100]         <1>       add    ebx, HF_TBL_VEC ; 21/02/2015
                                <1>       ;push  word [bx+2]        ; dpt segment
                                <1>       ;pop   es
                                <1>       ;mov   bx, [bx]           ; dpt offset
00005B8B 8B1B                   <1>       mov    ebx, [ebx]
                                <1> ;GV_EXIT:
00005B8D C3                     <1>       RETn
                                <1>
                                <1> hdc1_int: ; 21/02/2015
                                <1> ;--- HARDWARE INT 76H -- ( IRQ LEVEL  14 ) ----------------------
                                <1> ;                                                :
                                <1> ;     FIXED DISK INTERRUPT ROUTINE                     :
                                <1> ;                                                :
                                <1> ;----------------------------------------------------------------
                                <1>
                                <1> ; 22/12/2014
                                <1> ; IBM PC-XT Model 286 System BIOS Source Code - DISK.ASM (HD_INT)
                                <1> ;      '11/15/85'
                                <1> ; AWARD BIOS 1999 (D1A0622)
                                <1> ;     Source Code - ATORGS.ASM (INT_HDISK, INT_HDISK1)
                                <1>
                                <1> ;int_76h:
                                <1> HD_INT:
00005B8E 6650                   <1>       PUSH   AX
```

```
5394 00005B90 1E                  <1>       PUSH   DS
5395                              <1>       ;CALL  DDS
5396                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
5397 00005B91 66B81000            <1>       mov    ax, KDATA
5398 00005B95 8ED8                <1>       mov    ds, ax
5399                              <1>       ;
5400                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
5401                              <1>         ;mov     byte [CS:HF_INT_FLAG], 0FFh
5402 00005B97 C605[AB8A0100]FF    <1>       mov    byte [HF_INT_FLAG], 0FFh
5403                              <1>       ;
5404 00005B9E 6652                <1>       push   dx
5405 00005BA0 66BAF701            <1>       mov    dx, HDC1_BASEPORT+7 ; Status Register (1F7h)
5406                              <1>                                  ; Clear Controller
5407                              <1> Clear_IRQ1415:                        ; (Award BIOS - 1999)
5408 00005BA4 EC                  <1>       in     al, dx              ;
5409 00005BA5 665A                <1>       pop    dx
5410                              <1>       NEWIODELAY
5410 00005BA7 E6EB                <2>  out 0ebh,al
5411                              <1>       ;
5412 00005BA9 B020                <1>       MOV    AL,EOI              ; NON-SPECIFIC END OF INTERRUPT
5413 00005BAB E6A0                <1>       OUT    INTB00,AL           ; FOR CONTROLLER #2
5414                              <1>       ;JMP   $+2                 ; WAIT
5415                              <1>       NEWIODELAY
5415 00005BAD E6EB                <2>  out 0ebh,al
5416 00005BAF E620                <1>       OUT    INTA00,AL           ; FOR CONTROLLER #1
5417 00005BB1 1F                  <1>       POP    DS
5418                              <1>       ;STI                       ; RE-ENABLE INTERRUPTS
5419                              <1>       ;MOV   AX,9100H            ; DEVICE POST
5420                              <1>       ;INT   15H                 ;  INTERRUPT
5421                              <1> irq15_iret: ; 25/02/2015
5422 00005BB2 6658                <1>       POP    AX
5423 00005BB4 CF                  <1>       IRETd                      ; RETURN FROM INTERRUPT
5424                              <1>
5425                              <1> hdc2_int: ; 21/02/2015
5426                              <1> ;++++ HARDWARE INT 77H ++ ( IRQ LEVEL  15 ) ++++++++++++++++++++
5427                              <1> ;                                                  :
5428                              <1> ;     FIXED DISK INTERRUPT ROUTINE                       :
5429                              <1> ;                                                  :
5430                              <1> ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
5431                              <1>
5432                              <1> ;int_77h:
5433                              <1> HD1_INT:
5434 00005BB5 6650                <1>       PUSH   AX
5435                              <1>       ; Check if that is a spurious IRQ (from slave PIC)
5436                              <1>       ; 25/02/2015 (source: http://wiki.osdev.org/8259_PIC)
5437 00005BB7 B00B                <1>       mov    al, 0Bh  ; In-Service Register
5438 00005BB9 E6A0                <1>       out    0A0h, al
5439 00005BBB EB00                <1>         jmp short $+2
5440 00005BBD EB00                <1>       jmp short $+2
5441 00005BBF E4A0                <1>       in     al, 0A0h
5442 00005BC1 2480                <1>       and    al, 80h ; bit 7 (is it real IRQ 15 or fake?)
5443 00005BC3 74ED                <1>       jz     short irq15_iret ; Fake (spurious)IRQ, do not send EOI)
5444                              <1>       ;
5445 00005BC5 1E                  <1>       PUSH   DS
5446                              <1>       ;CALL  DDS
5447                              <1>       ; 21/02/2015 (32 bit, 386 pm modification)
5448 00005BC6 66B81000            <1>       mov    ax, KDATA
5449 00005BCA 8ED8                <1>       mov    ds, ax
5450                              <1>       ;
5451                              <1>       ;;MOV  @HF_INT_FLAG,0FFH   ; ALL DONE
5452                              <1>         ;or      byte [CS:HF_INT_FLAG],0C0h
5453 00005BCC 800D[AB8A0100]C0    <1>       or     byte [HF_INT_FLAG], 0C0h
5454                              <1>       ;
5455 00005BD3 6652                <1>       push   dx
5456 00005BD5 66BA7701            <1>       mov    dx, HDC2_BASEPORT+7 ; Status Register (177h)
5457                              <1>                                  ; Clear Controller (Award BIOS 1999)
5458 00005BD9 EBC9                <1>       jmp    short Clear_IRQ1415
5459                              <1>
5460                              <1>
5461                              <1> ;%include 'diskdata.inc' ; 11/03/2015
5462                              <1> ;%include 'diskbss.inc' ; 11/03/2015
5463                              <1>
5464                              <1>
5465                              <1> ;////////////////////////////////////////////////////////////////
5466                              <1> ;; END OF DISK I/O SYTEM ///
2895                                  %include 'memory.s'  ; 09/03/2015
   1                              <1> ; ****************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - memory.s
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 15/12/2020
   5                              <1> ; ----------------------------------------------------------------------------
   6                              <1> ; Beginning: 24/01/2016
   7                              <1> ; ----------------------------------------------------------------------------
   8                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
   9                              <1> ; ----------------------------------------------------------------------------
  10                              <1> ; Turkish Rational DOS
  11                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                              <1> ;
  13                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  14                              <1> ; memory.inc (18/10/2015)
  15                              <1> ; ****************************************************************************
  16                              <1>
  17                              <1> ; MEMORY.ASM - Retro UNIX 386 v1 MEMORY MANAGEMENT FUNCTIONS (PROCEDURES)
  18                              <1> ; Retro UNIX 386 v1 Kernel (unix386.s, v0.2.0.14) - MEMORY.INC
  19                              <1> ; Last Modification: 18/10/2015
  20                              <1>
  21                              <1> ; ///////// MEMORY MANAGEMENT FUNCTIONS (PROCEDURES) ///////////////
  22                              <1>
  23                              <1> ;;04/11/2014 (unix386.s)
  24                              <1> ;PDE_A_PRESENT     equ   1            ; Present flag for PDE
  25                              <1> ;PDE_A_WRITE equ   2            ; Writable (write permission) flag
  26                              <1> ;PDE_A_USER  equ   4            ; User (non-system/kernel) page flag
  27                              <1> ;;
  28                              <1> ;PTE_A_PRESENT     equ   1            ; Present flag for PTE (bit 0)
  29                              <1> ;PTE_A_WRITE equ   2            ; Writable (write permission) flag (bit 1)
```

```
30                                      <1> ;PTE_A_USER  equ    4           ; User (non-system/kernel) page flag (bit 2)
31                                      <1> ;PTE_A_ACCESS   equ      32          ; Accessed flag (bit 5) ; 09/03/2015
32                                      <1>
33                                      <1> ; 27/04/2015
34                                      <1> ; 09/03/2015
35                                      <1> PAGE_SIZE    equ 4096           ; page size in bytes
36                                      <1> PAGE_SHIFT   equ 12             ; page table shift count
37                                      <1> PAGE_D_SHIFT       equ 22 ; 12 + 10    ; page directory shift count
38                                      <1> PAGE_OFF     equ 0FFFh          ; 12 bit byte offset in page frame
39                                      <1> PTE_MASK     equ 03FFh          ; page table entry mask
40                                      <1> PTE_DUPLICATED  equ 200h        ; duplicated page sign (AVL bit 0)
41                                      <1> PDE_A_CLEAR  equ 0F000h         ; to clear PDE attribute bits
42                                      <1> PTE_A_CLEAR  equ 0F000h         ; to clear PTE attribute bits
43                                      <1> LOGIC_SECT_SIZE equ 512             ; logical sector size
44                                      <1> ERR_MAJOR_PF equ 0E0h           ; major error: page fault
45                                      <1> ERR_MINOR_IM equ 4 ;15/10/2016 (1->4); insufficient (out of) memory
46                                      <1> ERR_MINOR_PV equ 6 ;15/10/2016 (1->4); protection violation
47                                      <1> SWP_DISK_READ_ERR     equ 40
48                                      <1> SWP_DISK_NOT_PRESENT_ERR   equ 41
49                                      <1> SWP_SECTOR_NOT_PRESENT_ERR equ 42
50                                      <1> SWP_NO_FREE_SPACE_ERR      equ 43
51                                      <1> SWP_DISK_WRITE_ERR         equ 44
52                                      <1> SWP_NO_PAGE_TO_SWAP_ERR    equ 45
53                                      <1> PTE_A_ACCESS_BIT equ 5  ; Bit 5 (accessed flag)
54                                      <1> SECTOR_SHIFT     equ 3  ; sector shift (to convert page block number)
55                                      <1> ; 12/07/2016
56                                      <1> PTE_SHARED   equ 400h           ; AVL bit 1, direct memory access bit
57                                      <1>                                 ; (Indicates that the page is not allocated
58                                      <1>                                 ; for the process, it is a shared or system
59                                      <1>                                        ; page, it must not be deallocated!)
60                                      <1> ; 14/12/2020
61                                      <1> ; (Linear Frame Buffer - video memory mark : AVL bit 1, outside M.A.T.)
62                                      <1> PDE_EXTERNAL equ 400h     ; Page directory entry for external memory blocks
63                                      <1> PTE_EXTERNAL equ 400h     ; Allocated kernel pages for Linear Frame Buffer
64                                      <1>                           ; (Out of memory allocation table)
65                                      <1>
66                                      <1> ;
67                                      <1> ;; Retro Unix 386 v1 - paging method/principles
68                                      <1> ;;
69                                      <1> ;; 10/10/2014
70                                      <1> ;; RETRO UNIX 386 v1 - PAGING METHOD/PRINCIPLES
71                                      <1> ;;
72                                      <1> ;; KERNEL PAGE MAP: 1 to 1 physical memory page map
73                                      <1> ;;    (virtual address = physical address)
74                                      <1> ;; KERNEL PAGE TABLES:
75                                      <1> ;;    Kernel page directory and all page tables are
76                                      <1> ;;    on memory as initialized, as equal to physical memory
77                                      <1> ;;    layout. Kernel pages can/must not be swapped out/in.
78                                      <1> ;;
79                                      <1> ;;    what for: User pages may be swapped out, when accessing
80                                      <1> ;;    a page in kernel/system mode, if it would be swapped out,
81                                      <1> ;;    kernel would have to swap it in! But it is also may be
82                                      <1> ;;    in use by a user process. (In system/kernel mode
83                                      <1> ;;    kernel can access all memory pages even if they are
84                                      <1> ;;    reserved/allocated for user processes. Swap out/in would
85                                      <1> ;;    cause conflicts.)
86                                      <1> ;;
87                                      <1> ;;    As result of these conditions,
88                                      <1> ;;    all kernel pages must be initialized as equal to
89                                      <1> ;;    physical layout for preventing page faults.
90                                      <1> ;;    Also, calling "allocate page" procedure after
91                                      <1> ;;    a page fault can cause another page fault (double fault)
92                                      <1> ;;    if all kernel page tables would not be initialized.
93                                      <1> ;;
94                                      <1> ;;    [first_page] = Beginning of users space, as offset to
95                                      <1> ;;    memory allocation table. (double word aligned)
96                                      <1> ;;
97                                      <1> ;;    [next_page] = first/next free space to be searched
98                                      <1> ;;    as offset to memory allocation table. (dw aligned)
99                                      <1> ;;
100                                     <1> ;;    [last_page] = End of memory (users space), as offset
101                                     <1> ;;    to memory allocation table. (double word aligned)
102                                     <1> ;;
103                                     <1> ;; USER PAGE TABLES:
104                                     <1> ;;    Demand paging (& 'copy on write' allocation method) ...
105                                     <1> ;;           'ready only' marked copies of the
106                                     <1> ;;           parent process's page table entries (for
107                                     <1> ;;           same physical memory).
108                                     <1> ;;           (A page will be copied to a new page after
109                                     <1> ;;            if it causes R/W page fault.)
110                                     <1> ;;
111                                     <1> ;;    Every user process has own (different)
112                                     <1> ;;    page directory and page tables.
113                                     <1> ;;
114                                     <1> ;;    Code starts at virtual address 0, always.
115                                     <1> ;;    (Initial value of EIP is 0 in user mode.)
116                                     <1> ;;    (Programs can be written/developed as simple
117                                     <1> ;;     flat memory programs.)
118                                     <1> ;;
119                                     <1> ;; MEMORY ALLOCATION STRATEGY:
120                                     <1> ;;    Memory page will be allocated by kernel only
121                                     <1> ;;           (in kernel/system mode only).
122                                     <1> ;;    * After a
123                                     <1> ;;      - 'not present' page fault
124                                     <1> ;;      - 'writing attempt on read only page' page fault
125                                     <1> ;;    * For loading (opening, reading) a file or disk/drive
126                                     <1> ;;    * As responce to 'allocate additional memory blocks'
127                                     <1> ;;      request by running process.
128                                     <1> ;;    * While creating a process, allocating a new buffer,
129                                     <1> ;;      new page tables etc.
130                                     <1> ;;
131                                     <1> ;;    At first,
132                                     <1> ;;    - 'allocate page' procedure will be called;
133                                     <1> ;,       if it will return with a valid (>0) physical address
134                                     <1> ;;       (that means the relevant M.A.T. bit has been RESET)
```

```
<1> ;;       relevant memory page/block will be cleared (zeroed).
<1> ;;    - 'allocate page' will be called for allocating page
<1> ;;       directory, page table and running space (data/code).
<1> ;;    - every successful 'allocate page' call will decrease
<1> ;;      'free_pages' count (pointer).
<1> ;;    - 'out of (insufficient) memory error' will be returned
<1> ;;      if 'free_pages' points to a ZERO.
<1> ;;    - swapping out and swapping in (if it is not a new page)
<1> ;;      procedures will be called as responce to 'out of memory'
<1> ;;      error except errors caused by attribute conflicts.
<1> ;;     (swapper functions)
<1> ;;
<1> ;;    At second,
<1> ;;    - page directory entry will be updated then page table
<1> ;;      entry will be updated.
<1> ;;
<1> ;; MEMORY ALLOCATION TABLE FORMAT:
<1> ;;    - M.A.T. has a size according to available memory as
<1> ;;      follows:
<1> ;;            - 1 (allocation) bit per 1 page (4096 bytes)
<1> ;;            - a bit with value of 0 means allocated page
<1> ;;            - a bit with value of 1 means a free page
<1> ;,    - 'free_pages' pointer holds count of free pages
<1> ;;      depending on M.A.T.
<1> ;;          (NOTE: Free page count will not be checked
<1> ;;          again -on M.A.T.- after initialization.
<1> ;;          Kernel will trust on initial count.)
<1> ;,    - 'free_pages' count will be decreased by allocation
<1> ;;      and it will be increased by deallocation procedures.
<1> ;;
<1> ;;    - Available memory will be calculated during
<1> ;;      the kernel's initialization stage (in real mode).
<1> ;;      Memory allocation table and kernel page tables
<1> ;;      will be formatted/sized as result of available
<1> ;;      memory calculation before paging is enabled.
<1> ;;
<1> ;; For 4GB Available/Present Memory: (max. possible memory size)
<1> ;;    - Memory Allocation Table size will be 128 KB.
<1> ;;    - Memory allocation for kernel page directory size
<1> ;;      is always 4 KB. (in addition to total allocation size
<1> ;;      for page tables)
<1> ;;    - Memory allocation for kernel page tables (1024 tables)
<1> ;;      is 4 MB (1024*4*1024 bytes).
<1> ;;    - User (available) space will be started
<1> ;;      at 6th MB of the memory (after 1MB+4MB).
<1> ;;    - The first 640 KB is for kernel's itself plus
<1> ;;      memory allocation table and kernel's page directory
<1> ;;      (D0000h-EFFFFh may be used as kernel space...)
<1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
<1> ;;      for buffers.
<1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
<1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
<1> ;;    - Kernel page tables start at 100000h (2nd MB)
<1> ;;
<1> ;; For 1GB Available Memory:
<1> ;;    - Memory Allocation Table size will be 32 KB.
<1> ;;    - Memory allocation for kernel page directory size
<1> ;;      is always 4 KB. (in addition to total allocation size
<1> ;;      for page tables)
<1> ;;    - Memory allocation for kernel page tables (256 tables)
<1> ;;      is 1 MB (256*4*1024 bytes).
<1> ;;    - User (available) space will be started
<1> ;;      at 3th MB of the memory (after 1MB+1MB).
<1> ;;    - The first 640 KB is for kernel's itself plus
<1> ;;      memory allocation table and kernel's page directory
<1> ;;      (D0000h-EFFFFh may be used as kernel space...)
<1> ;;    - B0000h to B7FFFh address space (32 KB) will be used
<1> ;;      for buffers.
<1> ;;    - ROMBIOS, VIDEO BUFFER and VIDEO ROM space are reserved.
<1> ;,      (A0000h-AFFFFh, C0000h-CFFFFh, F0000h-FFFFFh)
<1> ;;    - Kernel page tables start at 100000h (2nd MB).
<1> ;;
<1> ;;
<1>
<1>
<1> ;;*****************************************************************************
<1> ;;
<1> ;; RETRO UNIX 386 v1 - Paging (Method for Copy On Write paging principle)
<1> ;; DEMAND PAGING - PARENT&CHILD PAGE TABLE DUPLICATION PRINCIPLES (23/04/2015)
<1>
<1> ;; Main factor: "sys fork" system call
<1> ;;
<1> ;;          FORK
<1> ;;                        |----> parent - duplicated PTEs, read only pages
<1> ;;  writable pages ---->|
<1> ;;                        |----> child - duplicated PTEs, read only pages
<1> ;;
<1> ;; AVL bit (0) of Page Table Entry is used as duplication sign
<1> ;;
<1> ;; AVL Bit 0 [PTE Bit 9] = 'Duplicated PTE belongs to child' sign/flag (if it is set)
<1> ;; Note: Dirty bit (PTE bit 6) may be used instead of AVL bit 0 (PTE bit 9)
<1> ;;       -while R/W bit is 0-.
<1> ;;
<1> ;; Duplicate page tables with writable pages (the 1st sys fork in the process):
<1> ;; # Parent's Page Table Entries are updated to point same pages as read only,
<1> ;;   as duplicated PTE bit  -AVL bit 0, PTE bit 9- are reset/clear.
<1> ;; # Then Parent's Page Table is copied to Child's Page Table.
<1> ;; # Child's Page Table Entries are updated as duplicated child bit
<1> ;;   -AVL bit 0, PTE bit 9- is set.
<1> ;;
<1> ;; Duplicate page tables with read only pages (several sys fork system calls):
<1> ;; # Parent's read only pages are copied to new child pages.
<1> ;;   Parent's PTE attributes are not changed.
<1> ;;   (Because, there is another parent-child fork before this fork! We must not
<1> ;;    destroy/mix previous fork result).
```

```
                        <1> ;; # Child's Page Table Entries (which are corresponding to Parent's
                        <1> ;;   read only pages) are set as writable (while duplicated PTE bit is clear).
                        <1> ;; # Parent's PTEs with writable page attribute are updated to point same pages
                        <1> ;;   as read only, (while) duplicated PTE bit is reset (clear).
                        <1> ;; # Parent's Page Table Entries (with writable page attribute) are duplicated
                        <1> ;;   as Child's Page Table Entries without copying actual page.
                        <1> ;; # Child 's Page Table Entries (which are corresponding to Parent's writable
                        <1> ;;   pages) are updated as duplicated PTE bit (AVL bit 0, PTE bit 9- is set.
                        <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>       ;
00005BDB A1[208A0100]   <1>       mov   eax, [free_pages]
00005BE0 21C0           <1>       and   eax, eax
00005BE2 7438           <1>       jz    short out_of_memory
                        <1>       ;
00005BE4 53             <1>       push  ebx
00005BE5 51             <1>       push  ecx
                        <1>       ;
00005BE6 BB00001000     <1>       mov   ebx, MEM_ALLOC_TBL   ; Memory Allocation Table offset
00005BEB 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)
00005BED 031D[248A0100] <1>       add   ebx, [next_page] ; Free page searching starts from here
                        <1>                             ; next_free_page >> 5
00005BF3 030D[288A0100] <1>       add   ecx, [last_page] ; Free page searching ends here
                        <1>                             ; (total_pages - 1) >> 5
                        <1> al_p_scan:
00005BF9 39CB           <1>       cmp   ebx, ecx
00005BFB 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>       ;
00005BFD 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.
00005C00 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!)
00005C02 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.
00005C05 EBF2           <1>         jmp     short al_p_scan
                        <1>       ;
                        <1> al_p_notfound:
00005C07 81E900001000   <1>       sub   ecx, MEM_ALLOC_TBL
00005C0D 890D[248A0100] <1>       mov   [next_page], ecx ; next/first free page = last page
                        <1>                              ; (deallocate_page procedure will change it)
00005C13 31C0           <1>       xor   eax, eax
00005C15 A3[208A0100]   <1>       mov   [free_pages], eax ; 0
00005C1A 59             <1>       pop   ecx
00005C1B 5B             <1>       pop   ebx
                        <1>       ;
                        <1> out_of_memory:
00005C1C E85B040000     <1>       call  swap_out
00005C21 7325           <1>       jnc   short al_p_ok  ; [free_pages] = 0, re-allocation by swap_out
                        <1>       ;
00005C23 29C0           <1>       sub   eax, eax ; 0
00005C25 F9             <1>       stc
00005C26 C3             <1>       retn
                        <1>
                        <1> al_p_found:
00005C27 89D9           <1>       mov   ecx, ebx
00005C29 81E900001000   <1>       sub   ecx, MEM_ALLOC_TBL
00005C2F 890D[248A0100] <1>       mov   [next_page], ecx ; Set first free page searching start
                        <1>                              ; address/offset (to the next)
00005C35 FF0D[208A0100] <1>         dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
                        <1>       ;
00005C3B 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)
00005C3E C1E103         <1>       shl   ecx, 3        ; (page block offset * 32) + page index
00005C41 01C8           <1>       add   eax, ecx      ; = page number
00005C43 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...
00005C46 59             <1>       pop   ecx
00005C47 5B             <1>       pop   ebx
                        <1> al_p_ok:
00005C48 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>       ;
00005C49 E88DFFFFFF     <1>       call   allocate_page
00005C4E 7216           <1>       jc     short mkpd_error
                        <1>       ;
00005C50 A3[B8030300]   <1>       mov    [u.pgdir], eax    ; Page dir address for current user/process
                        <1>                                ; (Physical address)
                        <1> clear_page:
                        <1>       ; 18/04/2015
                        <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;      EAX = physical address of the page
                        <1>       ; OUTPUT ->
                        <1>       ;      all bytes of the page will be cleared
                        <1>       ;
                        <1>       ; Modified Registers -> none
                        <1>       ;
00005C55 57             <1>       push   edi
00005C56 51             <1>       push   ecx
00005C57 50             <1>       push   eax
00005C58 B900040000     <1>       mov    ecx, PAGE_SIZE / 4
00005C5D 89C7           <1>       mov    edi, eax
00005C5F 31C0           <1>       xor    eax, eax
00005C61 F3AB           <1>       rep    stosd
00005C63 58             <1>       pop    eax
00005C64 59             <1>       pop    ecx
00005C65 5F             <1>       pop    edi
                        <1> mkpd_error:
                        <1> mkpt_error:
00005C66 C3             <1>       retn
                        <1>
                        <1> make_page_table:
                        <1>       ; 23/06/2015
                        <1>       ; 18/04/2015
                        <1>       ; 12/04/2015
                        <1>       ; 16/10/2014
                        <1>       ; 09/10/2014 ; (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;      EBX = virtual (linear) address
                        <1>       ;      ECX = page table attributes (lower 12 bits)
                        <1>       ;            (higher 20 bits must be ZERO)
                        <1>       ;            (bit 0 must be 1)
                        <1>       ;      u.pgdir = page directory (physical) address
                        <1>       ; OUTPUT ->
                        <1>       ;      EDX = Page directory entry address
                        <1>       ;      EAX = Page table address
                        <1>       ;      cf = 1 -> insufficient (out of) memory error
                        <1>       ;      cf = 0 -> page table address in the PDE (EDX)
                        <1>       ;
                        <1>       ; Modified Registers -> EAX, EDX
                        <1>       ;
00005C67 E86FFFFFFF     <1>       call   allocate_page
00005C6C 72F8           <1>       jc     short mkpt_error
00005C6E E811000000     <1>       call   set_pde
00005C73 EBE0           <1>       jmp    short clear_page
                        <1>
                        <1> make_page:
                        <1>       ; 24/07/2015
                        <1>       ; 23/06/2015 ; (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;      EBX = virtual (linear) address
                        <1>       ;      ECX = page attributes (lower 12 bits)
                        <1>       ;            (higher 20 bits must be ZERO)
                        <1>       ;            (bit 0 must be 1)
                        <1>       ;      u.pgdir = page directory (physical) address
                        <1>       ; OUTPUT ->
                        <1>       ;      EBX = Virtual address
                        <1>       ;      (EDX = PTE value)
                        <1>       ;      EAX = Physical address
                        <1>       ;      cf = 1 -> insufficient (out of) memory error
                        <1>       ;
                        <1>       ; Modified Registers -> EAX, EDX
                        <1>       ;
00005C75 E861FFFFFF     <1>       call   allocate_page
00005C7A 7207           <1>       jc     short mkp_err
00005C7C E821000000     <1>       call   set_pte
00005C81 73D2           <1>       jnc    short clear_page ; 18/04/2015
                        <1> mkp_err:
00005C83 C3             <1>       retn
                        <1>
                        <1>
                        <1> set_pde:     ; Set page directory entry (PDE)
                        <1>       ; 20/07/2015
                        <1>       ; 18/04/2015
                        <1>       ; 12/04/2015
                        <1>       ; 23/10/2014
                        <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;      EAX = physical address
                        <1>       ;            (use present value if EAX = 0)
                        <1>       ;      EBX = virtual (linear) address
                        <1>       ;      ECX = page table attributes (lower 12 bits)
                        <1>       ;            (higher 20 bits must be ZERO)
                        <1>       ;            (bit 0 must be 1)
                        <1>       ;      u.pgdir = page directory (physical) address
                        <1>       ; OUTPUT ->
                        <1>       ;      EDX = PDE address
                        <1>       ;      EAX = page table address (physical)
                        <1>       ;      ;(CF=1 -> Invalid page address)
```

```
                                <1>       ;
                                <1>       ; Modified Registers -> EDX
                                <1>       ;
00005C84 89DA                   <1>       mov    edx, ebx
00005C86 C1EA16                 <1>       shr    edx, PAGE_D_SHIFT ; 22
00005C89 C1E202                 <1>       shl    edx, 2 ; offset to page directory (1024*4)
00005C8C 0315[B8030300]         <1>       add    edx, [u.pgdir]
                                <1>       ;
00005C92 21C0                   <1>       and    eax, eax
00005C94 7506                   <1>       jnz    short spde_1
                                <1>       ;
00005C96 8B02                   <1>       mov    eax, [edx]  ; old PDE value
                                <1>       ;test  al, 1
                                <1>       ;jz    short spde_2
00005C98 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h  ; clear lower 12 bits
                                <1> spde_1:
                                <1>       ;and   cx, 0FFFh
00005C9C 8902                   <1>       mov    [edx], eax
00005C9E 66090A                 <1>       or     [edx], cx
00005CA1 C3                     <1>       retn
                                <1> ;spde_2: ; error
                                <1> ;     stc
                                <1> ;     retn
                                <1>
                                <1> set_pte:    ; Set page table entry (PTE)
                                <1>       ; 24/07/2015
                                <1>       ; 20/07/2015
                                <1>       ; 23/06/2015
                                <1>       ; 18/04/2015
                                <1>       ; 12/04/2015
                                <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EAX = physical page address
                                <1>       ;            (use present value if EAX = 0)
                                <1>       ;      EBX = virtual (linear) address
                                <1>       ;      ECX = page attributes (lower 12 bits)
                                <1>       ;            (higher 20 bits must be ZERO)
                                <1>       ;            (bit 0 must be 1)
                                <1>       ;      u.pgdir = page directory (physical) address
                                <1>       ; OUTPUT ->
                                <1>       ;      EAX = physical page address
                                <1>       ;      (EDX = PTE value)
                                <1>       ;      EBX = virtual address
                                <1>       ;
                                <1>       ;      CF = 1 -> error
                                <1>       ;
                                <1>       ; Modified Registers -> EAX, EDX
                                <1>       ;
00005CA2 50                     <1>       push   eax
00005CA3 A1[B8030300]           <1>       mov    eax, [u.pgdir] ; 20/07/2015
00005CA8 E837000000             <1>       call   get_pde
                                <1>              ; EDX = PDE address
                                <1>              ; EAX = PDE value
00005CAD 5A                     <1>       pop    edx ; physical page address
00005CAE 722A                   <1>       jc     short spte_err ; PDE not present
                                <1>       ;
00005CB0 53                     <1>       push   ebx ; 24/07/2015
00005CB1 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
                                <1>                       ; EDX = PT address (physical)
00005CB5 C1EB0C                 <1>       shr    ebx, PAGE_SHIFT ; 12
00005CB8 81E3FF030000           <1>       and    ebx, PTE_MASK; 03FFh
                                <1>                      ; clear higher 10 bits (PD bits)
00005CBE C1E302                 <1>       shl    ebx, 2   ; offset to page table (1024*4)
00005CC1 01C3                   <1>       add    ebx, eax
                                <1>       ;
00005CC3 8B03                   <1>       mov    eax, [ebx] ; Old PTE value
00005CC5 A801                   <1>       test   al, 1
00005CC7 740C                   <1>       jz     short spte_0
00005CC9 09D2                   <1>       or     edx, edx
00005CCB 750F                   <1>       jnz    short spte_1
00005CCD 662500F0               <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 bits
00005CD1 89C2                   <1>       mov    edx, eax
00005CD3 EB09                   <1>       jmp    short spte_2
                                <1> spte_0:
                                <1>       ; If this PTE contains a swap (disk) address,
                                <1>       ; it can be updated by using 'swap_in' procedure
                                <1>       ; only!
00005CD5 21C0                   <1>       and    eax, eax
00005CD7 7403                   <1>       jz     short spte_1
                                <1>       ; 24/07/2015
                                <1>       ; swapped page ! (on disk)
00005CD9 5B                     <1>       pop    ebx
                                <1> spte_err:
00005CDA F9                     <1>       stc
00005CDB C3                     <1>       retn
                                <1> spte_1:
00005CDC 89D0                   <1>       mov    eax, edx
                                <1> spte_2:
00005CDE 09CA                   <1>       or     edx, ecx
                                <1>       ; 23/06/2015
00005CE0 8913                   <1>       mov    [ebx], edx ; PTE value in EDX
                                <1>       ; 24/07/2015
00005CE2 5B                     <1>       pop    ebx
00005CE3 C3                     <1>       retn
                                <1>
                                <1> get_pde:    ; Get present value of the relevant PDE
                                <1>       ; 20/07/2015
                                <1>       ; 18/04/2015
                                <1>       ; 12/04/2015
                                <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EBX = virtual (linear) address
                                <1>       ;      EAX = page directory (physical) address
```

```
                        <1>       ; OUTPUT ->
                        <1>       ;      EDX = Page directory entry address
                        <1>       ;      EAX = Page directory entry value
                        <1>       ;      CF = 1 -> PDE not present or invalid ?
                        <1>       ; Modified Registers -> EDX, EAX
                        <1>       ;
00005CE4 89DA           <1>       mov    edx, ebx
00005CE6 C1EA16         <1>       shr    edx, PAGE_D_SHIFT ; 22  (12+10)
00005CE9 C1E202         <1>       shl    edx, 2 ; offset to page directory (1024*4)
00005CEC 01C2           <1>       add    edx, eax ; page directory address (physical)
00005CEE 8B02           <1>       mov    eax, [edx]
00005CF0 A801           <1>       test   al, PDE_A_PRESENT ; page table is present or not !
00005CF2 751F           <1>       jnz    short gpte_retn
00005CF4 F9             <1>       stc
                        <1> gpde_retn:
00005CF5 C3             <1>       retn
                        <1>
                        <1> get_pte:
                        <1>               ; Get present value of the relevant PTE
                        <1>       ; 29/07/2015
                        <1>       ; 20/07/2015
                        <1>       ; 18/04/2015
                        <1>       ; 12/04/2015
                        <1>       ; 10/10/2014 ; (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;      EBX = virtual (linear) address
                        <1>       ;      EAX = page directory (physical) address
                        <1>       ; OUTPUT ->
                        <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 ?
                        <1>       ; Modified Registers -> EAX, EDX
                        <1>       ;
00005CF6 E8E9FFFFFF     <1>       call   get_pde
00005CFB 72F8           <1>       jc     short gpde_retn     ; page table is not present
                        <1>       ;jnc   short gpte_1
                        <1>       ;retn
                        <1> ;gpte_1:
00005CFD 662500F0       <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 bits
00005D01 89DA           <1>       mov    edx, ebx
00005D03 C1EA0C         <1>       shr    edx, PAGE_SHIFT ; 12
00005D06 81E2FF030000   <1>       and    edx, PTE_MASK; 03FFh
                        <1>                       ; clear higher 10 bits (PD bits)
00005D0C C1E202         <1>       shl    edx, 2 ; offset from start of page table (1024*4)
00005D0F 01C2           <1>       add    edx, eax
00005D11 8B02           <1>       mov    eax, [edx]
                        <1> gpte_retn:
00005D13 C3             <1>       retn
                        <1>
                        <1> deallocate_page_dir:
                        <1>       ; 15/09/2015
                        <1>       ; 05/08/2015
                        <1>       ; 30/04/2015
                        <1>       ; 28/04/2015
                        <1>       ; 17/10/2014
                        <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;      EAX = PHYSICAL ADDRESS OF THE PAGE DIRECTORY (CHILD)
                        <1>       ;      EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
                        <1>       ; OUTPUT ->
                        <1>       ;      All of page tables in the page directory
                        <1>       ;      and page dir's itself will be deallocated
                        <1>       ;      except 'read only' duplicated pages (will be converted
                        <1>       ;      to writable pages).
                        <1>       ;
                        <1>       ; Modified Registers -> EAX
                        <1>       ;
                        <1>       ;
00005D14 56             <1>       push   esi
00005D15 51             <1>       push   ecx
00005D16 50             <1>       push   eax
00005D17 89C6           <1>       mov    esi, eax
00005D19 31C9           <1>       xor    ecx, ecx
                        <1>       ; The 1st PDE points to Kernel Page Table 0 (the 1st 4MB),
                        <1>       ; it must not be deallocated
00005D1B 890E           <1>       mov    [esi], ecx ; 0 ; clear PDE 0
                        <1> dapd_0:
00005D1D AD             <1>       lodsd
00005D1E A801           <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
00005D20 7409           <1>       jz     short dapd_1
00005D22 662500F0       <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
00005D26 E812000000     <1>       call   deallocate_page_table
                        <1> dapd_1:
00005D2B 41             <1>       inc    ecx ; page directory entry index
00005D2C 81F900040000   <1>       cmp    ecx, PAGE_SIZE / 4 ; 1024
00005D32 72E9           <1>       jb     short dapd_0
                        <1> dapd_2:
00005D34 58             <1>       pop    eax
00005D35 E87F000000     <1>       call   deallocate_page     ; deallocate the page dir's itself
00005D3A 59             <1>       pop    ecx
00005D3B 5E             <1>       pop    esi
00005D3C C3             <1>       retn
                        <1>
                        <1> deallocate_page_table:
                        <1>       ; 12/07/2016
                        <1>       ; 19/09/2015
                        <1>       ; 15/09/2015
                        <1>       ; 05/08/2015
                        <1>       ; 30/04/2015
                        <1>       ; 28/04/2015
                        <1>       ; 24/10/2014
```

```
                        <1>       ; 23/10/2014
                        <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE PAGE TABLE
                        <1>       ;     EBX = PHYSICAL ADDRESS OF THE PARENT'S PAGE DIRECTORY
                        <1>       ;     (ECX = page directory entry index)
                        <1>       ; OUTPUT ->
                        <1>       ;     All of pages in the page table and page table's itself
                        <1>       ;     will be deallocated except 'read only' duplicated pages
                        <1>       ;     (will be converted to writable pages).
                        <1>       ;
                        <1>       ; Modified Registers -> EAX
                        <1>       ;
00005D3D 56             <1>       push   esi
00005D3E 57             <1>       push   edi
00005D3F 52             <1>       push   edx
00005D40 50             <1>       push   eax ; *
00005D41 89C6           <1>       mov    esi, eax
00005D43 31FF           <1>       xor    edi, edi ; 0
                        <1> dapt_0:
00005D45 AD             <1>       lodsd
00005D46 A801           <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
00005D48 7441           <1>       jz     short dapt_1
                        <1>       ;
00005D4A A802           <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                        <1>                                ; (must be 1)
00005D4C 754C           <1>       jnz    short dapt_3
                        <1>       ; Read only -duplicated- page (belongs to a parent or a child)
00005D4E 66A90002       <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                        <1>                                    ; as child's page ?
00005D52 7451           <1>       jz     short dapt_4 ; Clear PTE but don't deallocate the page!
                        <1>       ; check the parent's PTE value is read only & same page or not..
                        <1>       ; ECX = page directory entry index (0-1023)
00005D54 53             <1>       push   ebx
00005D55 51             <1>       push   ecx
00005D56 66C1E102       <1>       shl    cx, 2 ; *4
00005D5A 01CB           <1>       add    ebx, ecx ; PDE offset (for the parent)
00005D5C 8B0B           <1>       mov    ecx, [ebx]
00005D5E F6C101         <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
00005D61 7435           <1>       jz     short dapt_2 ; parent process does not use this page
00005D63 6681E100F0     <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
                        <1>       ; EDI = page table entry index (0-1023)
00005D68 89FA           <1>       mov    edx, edi
00005D6A 66C1E202       <1>       shl    dx, 2 ; *4
00005D6E 01CA           <1>       add    edx, ecx ; PTE offset (for the parent)
00005D70 8B1A           <1>       mov    ebx, [edx]
00005D72 F6C301         <1>       test   bl, PTE_A_PRESENT ; present or not ?
00005D75 7421           <1>       jz     short dapt_2 ; parent process does not use this page
00005D77 662500F0       <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00005D7B 6681E300F0     <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00005D80 39D8           <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
00005D82 7514           <1>       jne    short dapt_2 ; not same page
                        <1>                           ; deallocate the child's page
00005D84 800A02         <1>         or     byte [edx], PTE_A_WRITE ; convert to writable page (parent)
00005D87 59             <1>       pop    ecx
00005D88 5B             <1>       pop    ebx
00005D89 EB1A           <1>       jmp    short dapt_4
                        <1> dapt_1:
00005D8B 09C0           <1>       or     eax, eax     ; swapped page ?
00005D8D 741D           <1>       jz     short dapt_5 ; no
                        <1>                           ; yes
00005D8F D1E8           <1>       shr    eax, 1
00005D91 E8CA040000     <1>       call   unlink_swap_block ; Deallocate swapped page block
                        <1>                                ; on the swap disk (or in file)
00005D96 EB14           <1>       jmp    short dapt_5
                        <1> dapt_2:
00005D98 59             <1>       pop    ecx
00005D99 5B             <1>       pop    ebx
                        <1> dapt_3:
                        <1>       ; 12/07/2016
00005D9A 66A90004       <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
00005D9E 7505           <1>       jnz    short dapt_4   ; AVL bit 1 = 1, do not deallocate this page!
                        <1>       ;
                        <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
00005DA0 E814000000     <1>       call   deallocate_page ; set the mem allocation bit of this page
                        <1> dapt_4:
00005DA5 C746FC00000000 <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
                        <1> dapt_5:
00005DAC 47             <1>       inc    edi ; page table entry index
00005DAD 81FF00040000   <1>       cmp    edi, PAGE_SIZE / 4 ; 1024
00005DB3 7290           <1>       jb     short dapt_0
                        <1>       ;
00005DB5 58             <1>       pop    eax ; *
00005DB6 5A             <1>       pop    edx
00005DB7 5F             <1>       pop    edi
00005DB8 5E             <1>       pop    esi
                        <1>       ;
                        <1>       ;call  deallocate_page     ; deallocate the page table's itself
                        <1>       ;retn
                        <1>
                        <1> deallocate_page:
                        <1>       ; 15/09/2015
                        <1>       ; 28/04/2015
                        <1>       ; 10/03/2015
                        <1>       ; 17/10/2014
                        <1>       ; 12/10/2014 (Retro UNIX 386 v1 - beginning)
                        <1>       ;
                        <1>       ; INPUT ->
                        <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
                        <1>       ; OUTPUT ->
                        <1>       ;     [free_pages] is increased
                        <1>       ;     (corresponding MEMORY ALLOCATION TABLE bit is SET)
                        <1>       ;     CF = 1 if the page is already deallocated
                        <1>       ;             (or not allocated) before.
```

```
                                    <1>         ;
                                    <1>         ; Modified Registers -> EAX
                                    <1>         ;
00005DB9 53                         <1>         push   ebx
00005DBA 52                         <1>         push   edx
                                    <1>         ;
00005DBB C1E80C                     <1>         shr    eax, PAGE_SHIFT      ; shift physical address to
                                    <1>                                 ; 12 bits right
                                    <1>                                 ; to get page number
00005DBE 89C2                       <1>         mov    edx, eax
                                    <1>         ; 15/09/2015
00005DC0 C1EA03                     <1>         shr    edx, 3           ; to get offset to M.A.T.
                                    <1>                                 ; (1 allocation bit = 1 page)
                                    <1>                                 ; (1 allocation bytes = 8 pages)
00005DC3 80E2FC                     <1>         and    dl, 0FCh         ; clear lower 2 bits
                                    <1>                                 ; (to get 32 bit position)
                                    <1>         ;
00005DC6 BB00001000                 <1>         mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address
00005DCB 01D3                       <1>         add    ebx, edx
00005DCD 83E01F                     <1>         and    eax, 1Fh         ; lower 5 bits only
                                    <1>                                 ; (allocation bit position)
00005DD0 3B15[248A0100]             <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)
00005DD6 7306                       <1>         jnb    short dap_1      ; no
00005DD8 8915[248A0100]             <1>         mov    [next_page], edx     ; yes
                                    <1> dap_1:
00005DDE 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.
00005DE1 FF05[208A0100]             <1>           inc      dword [free_pages]
                                    <1> dap_2:
00005DE7 5A                         <1>         pop    edx
00005DE8 5B                         <1>         pop    ebx
00005DE9 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>       ;
00005DEA E8ECFDFFFF             <1>       call   allocate_page
00005DEF 723E                   <1>       jc     short dpd_err
                                <1>       ;
00005DF1 55                     <1>       push   ebp ; 20/07/2015
00005DF2 56                     <1>       push   esi
00005DF3 57                     <1>       push   edi
00005DF4 53                     <1>       push   ebx
00005DF5 51                     <1>       push   ecx
00005DF6 8B35[B8030300]         <1>       mov    esi, [u.pgdir]
00005DFC 89C7                   <1>       mov    edi, eax
00005DFE 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)
00005DFF A5                     <1>       movsd
00005E00 BD00004000             <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
00005E05 B9FF030000             <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
                                <1> dpd_0:
00005E0A AD                     <1>       lodsd
                                <1>       ;or    eax, eax
                                <1>         ;jnz     short dpd_1
00005E0B A801                   <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
00005E0D 7508                   <1>       jnz    short dpd_1
                                <1>       ; 20/07/2015 (virtual address at the end of the page table)
00005E0F 81C500004000           <1>       add    ebp, 1024*4096 ; page size * PTE count
00005E15 EB0F                   <1>       jmp    short dpd_2
                                <1> dpd_1:
00005E17 662500F0               <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
00005E1B 89C3                   <1>       mov    ebx, eax
                                <1>       ; EBX = Parent's page table address
00005E1D E81F000000             <1>       call   duplicate_page_table
00005E22 720C                   <1>       jc     short dpd_p_err
                                <1>       ; EAX = Child's page table address
00005E24 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:
00005E26 AB                     <1>       stosd
00005E27 E2E1                   <1>       loop   dpd_0
                                <1>       ;
00005E29 58                     <1>       pop    eax  ; restore child's page directory address
                                <1> dpd_3:
00005E2A 59                     <1>       pop    ecx
00005E2B 5B                     <1>       pop    ebx
00005E2C 5F                     <1>       pop    edi
00005E2D 5E                     <1>       pop    esi
00005E2E 5D                     <1>       pop    ebp ; 20/07/2015
                                <1> dpd_err:
00005E2F C3                     <1>       retn
                                <1> dpd_p_err:
                                <1>       ; release the allocated pages missing (recover free space)
00005E30 58                     <1>       pop    eax  ; the new page directory address (physical)
00005E31 8B1D[B8030300]         <1>       mov    ebx, [u.pgdir] ; parent's page directory address
00005E37 E8D8FEFFFF             <1>       call   deallocate_page_dir
00005E3C 29C0                   <1>       sub    eax, eax ; 0
00005E3E F9                     <1>       stc
00005E3F 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>       ;
00005E41 E895FDFFFF             <1>       call   allocate_page
00005E46 726A                   <1>       jc     short dpt_err
                                <1>       ;
00005E48 50                     <1>       push   eax ; *
00005E49 56                     <1>       push   esi
```

```
1080 00005E4A 57                <1>       push   edi
1081 00005E4B 52                <1>       push   edx
1082 00005E4C 51                <1>       push   ecx
1083                            <1>       ;
1084 00005E4D 89DE              <1>       mov    esi, ebx
1085 00005E4F 89C7              <1>       mov    edi, eax
1086 00005E51 89C2              <1>       mov    edx, eax
1087 00005E53 81C200100000      <1>       add    edx, PAGE_SIZE
1088                            <1> dpt_0:
1089 00005E59 AD                <1>       lodsd
1090 00005E5A 21C0              <1>       and    eax, eax
1091 00005E5C 7444              <1>       jz     short dpt_3
1092 00005E5E A801              <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
1093 00005E60 7507              <1>       jnz    short dpt_1
1094                            <1>       ; 20/07/2015
1095                            <1>       ; ebp = virtual (linear) address of the memory page
1096 00005E62 E83F050000        <1>       call   reload_page ; 28/04/2015
1097 00005E67 7244              <1>       jc     short dpt_p_err
1098                            <1> dpt_1:
1099                            <1>       ; 21/09/2015
1100 00005E69 89C1              <1>       mov    ecx, eax
1101 00005E6B 662500F0          <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1102 00005E6F F6C102            <1>       test   cl, PTE_A_WRITE ; writable page ?
1103 00005E72 7525              <1>       jnz    short dpt_2
1104                            <1>       ; Read only (parent) page
1105                            <1>       ;      - there is a third process which uses this page -
1106                            <1>       ; Allocate a new page for the child process
1107 00005E74 E862FDFFFF        <1>       call   allocate_page
1108 00005E79 7232              <1>       jc     short dpt_p_err
1109 00005E7B 57                <1>       push   edi
1110 00005E7C 56                <1>       push   esi
1111 00005E7D 89CE              <1>       mov    esi, ecx
1112 00005E7F 89C7              <1>       mov    edi, eax
1113 00005E81 B900040000        <1>       mov    ecx, PAGE_SIZE/4
1114 00005E86 F3A5              <1>       rep    movsd  ; copy page (4096 bytes)
1115 00005E88 5E                <1>       pop    esi
1116 00005E89 5F                <1>       pop    edi
1117                            <1>       ;
1118 00005E8A 53                <1>       push   ebx
1119 00005E8B 50                <1>       push   eax
1120                            <1>       ; 20/07/2015
1121 00005E8C 89EB              <1>       mov    ebx, ebp
1122                            <1>       ; ebx = virtual (linear) address of the memory page
1123 00005E8E E887030000        <1>       call   add_to_swap_queue
1124 00005E93 58                <1>       pop    eax
1125 00005E94 5B                <1>       pop    ebx
1126                            <1>       ; 21/09/2015
1127 00005E95 0C07              <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
1128                            <1>              ; user + writable + present page
1129 00005E97 EB09              <1>       jmp    short dpt_3
1130                            <1> dpt_2:
1131                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
1132 00005E99 0C05              <1>       or     al, PTE_A_USER+PTE_A_PRESENT
1133                            <1>                  ; (read only page!)
1134 00005E9B 8946FC            <1>       mov    [esi-4], eax ; update parent's PTE
1135 00005E9E 660D0002          <1>       or      ax, PTE_DUPLICATED  ; (read only page & duplicated PTE!)
1136                            <1> dpt_3:
1137 00005EA2 AB                <1>       stosd  ; EDI points to child's PTE
1138                            <1>       ;
1139 00005EA3 81C500100000      <1>       add    ebp, 4096 ; 20/07/2015 (next page)
1140                            <1>       ;
1141 00005EA9 39D7              <1>       cmp    edi, edx
1142 00005EAB 72AC              <1>       jb     short dpt_0
1143                            <1> dpt_p_err:
1144 00005EAD 59                <1>       pop    ecx
1145 00005EAE 5A                <1>       pop    edx
1146 00005EAF 5F                <1>       pop    edi
1147 00005EB0 5E                <1>       pop    esi
1148 00005EB1 58                <1>       pop    eax ; *
1149                            <1> dpt_err:
1150 00005EB2 C3                <1>       retn
1151                            <1>
1152                            <1> page_fault_handler:      ; CPU EXCEPTION 0Eh (14) : Page Fault !
1153                            <1>       ; 21/09/2015
1154                            <1>       ; 19/09/2015
1155                            <1>       ; 17/09/2015
1156                            <1>       ; 28/08/2015
1157                            <1>       ; 20/07/2015
1158                            <1>       ; 28/06/2015
1159                            <1>       ; 03/05/2015
1160                            <1>       ; 30/04/2015
1161                            <1>       ; 18/04/2015
1162                            <1>       ; 12/04/2015
1163                            <1>       ; 30/10/2014
1164                            <1>       ; 11/09/2014
1165                            <1>       ; 10/09/2014 (Retro UNIX 386 v1 - beginning)
1166                            <1>       ;
1167                            <1>       ; Note: This is not an interrupt/exception handler.
1168                            <1>       ;     This is a 'page fault remedy' subroutine
1169                            <1>       ;     which will be called by standard/uniform
1170                            <1>       ;     exception handler.
1171                            <1>       ;
1172                            <1>       ; INPUT ->
1173                            <1>       ;     [error_code] = 32 bit ERROR CODE (lower 5 bits are valid)
1174                            <1>       ;
1175                            <1>       ;     cr2 = the virtual (linear) address
1176                            <1>       ;           which has caused to page fault (19/09/2015)
1177                            <1>       ;
1178                            <1>       ; OUTPUT ->
1179                            <1>       ;     (corresponding PAGE TABLE ENTRY is mapped/set)
1180                            <1>       ;     EAX = 0 -> no error
1181                            <1>       ;     EAX > 0 -> error code in EAX (also CF = 1)
1182                            <1>       ;
1183                            <1>       ; Modified Registers -> none (except EAX)
1184                            <1>       ;
```

```
1185                                    <1>         ;
1186                                    <1>         ; ERROR CODE:
1187                                    <1>       ;       31  .....   4   3  2   1  0
1188                                    <1>       ;      +---+-- --+---+---+---+---+---+
1189                                    <1>       ;      |   Reserved  | I | R | U | W | P |
1190                                    <1>       ;      +---+-- --+---+---+---+---+---+
1191                                    <1>       ;
1192                                    <1>       ; P : PRESENT -     When set, the page fault was caused by
1193                                    <1>       ;             a page-protection violation. When not set,
1194                                    <1>       ;             it was caused by a non-present page.
1195                                    <1>       ; W : WRITE   -     When set, the page fault was caused by
1196                                    <1>       ;             a page write. When not set, it was caused
1197                                    <1>       ;             by a page read.
1198                                    <1>       ; U : USER    -     When set, the page fault was caused
1199                                    <1>       ;             while CPL = 3.
1200                                    <1>       ;             This does not necessarily mean that
1201                                    <1>       ;             the page fault was a privilege violation.
1202                                    <1>       ; R : RESERVD -     When set, the page fault was caused by
1203                                    <1>       ;     WRITE   reading a 1 in a reserved field.
1204                                    <1>       ; I : INSTRUC -     When set, the page fault was caused by
1205                                    <1>       ;     FETCH   an instruction fetch
1206                                    <1>       ;
1207                                    <1>       ;; x86 (32 bit) VIRTUAL ADDRESS TRANSLATION
1208                                    <1>       ;  31               22                  12 11                      0
1209                                    <1>       ; +------------------+------------------+-----------------------+
1210                                    <1>             ; | PAGE DIR. ENTRY # | PAGE TAB. ENTRY # |        OFFSET         |
1211                                    <1>             ; +------------------+------------------+-----------------------+
1212                                    <1>       ;
1213                                    <1>
1214                                    <1>       ;; CR3 REGISTER (Control Register 3)
1215                                    <1>       ;  31                                    12             5 4 3 2   0
1216                                    <1>       ; +----------------------------------------+-------------+---+-----+
1217                                    <1>             ; |                                          |         |P|P|     |
1218                                    <1>             ; |   PAGE DIRECTORY TABLE BASE ADDRESS      |  reserved     |C|W|rsvrd|
1219                                    <1>             ; |                                          |         |D|T|     |
1220                                    <1>       ; +----------------------------------------+-------------+---+-----+
1221                                    <1>       ;
1222                                    <1>       ;     PWT    - WRITE THROUGH
1223                                    <1>       ;     PCD    - CACHE DISABLE
1224                                    <1>       ;
1225                                    <1>       ;
1226                                    <1>       ;; x86 PAGE DIRECTORY ENTRY (4 KByte Page)
1227                                    <1>       ;  31                                  12 11  9 8 7 6 5 4 3 2 1 0
1228                                    <1>       ; +--------------------------------------+-----+---+-+-+---+-+-+-+
1229                                    <1>             ; |                                        |     | | | | |P|P|U|R| |
1230                                    <1>             ; |     PAGE TABLE BASE ADDRESS 31..12     | AVL |G|0|D|A|C|W|/|/|P|
1231                                    <1>             ; |                                        |     | | | | |D|T|S|W| |
1232                                    <1>       ; +--------------------------------------+-----+---+-+-+---+-+-+-+
1233                                    <1>       ;
1234                                    <1>         ;       P      - PRESENT
1235                                    <1>         ;       R/W    - READ/WRITE
1236                                    <1>         ;       U/S    - USER/SUPERVISOR
1237                                    <1>       ;     PWT    - WRITE THROUGH
1238                                    <1>       ;     PCD    - CACHE DISABLE
1239                                    <1>       ;     A      - ACCESSED
1240                                    <1>         ;       D      - DIRTY (IGNORED)
1241                                    <1>       ;     PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
1242                                    <1>       ;     G      - GLOBAL     (IGNORED)
1243                                    <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
1244                                    <1>       ;
1245                                    <1>       ;
1246                                    <1>       ;; x86 PAGE TABLE ENTRY (4 KByte Page)
1247                                    <1>       ;  31                                  12 11  9 8 7 6 5 4 3 2 1 0
1248                                    <1>       ; +--------------------------------------+-----+---+-+-+---+-+-+-+
1249                                    <1>             ; |                                        |     | |P| | |P|P|U|R| |
1250                                    <1>             ; |     PAGE FRAME BASE ADDRESS 31..12     | AVL |G|A|D|A|C|W|/|/|P|
1251                                    <1>             ; |                                        |     | |T| | |D|T|S|W| |
1252                                    <1>       ; +--------------------------------------+-----+---+-+-+---+-+-+-+
1253                                    <1>       ;
1254                                    <1>         ;       P      - PRESENT
1255                                    <1>         ;       R/W    - READ/WRITE
1256                                    <1>         ;       U/S    - USER/SUPERVISOR
1257                                    <1>       ;     PWT    - WRITE THROUGH
1258                                    <1>       ;     PCD    - CACHE DISABLE
1259                                    <1>       ;     A      - ACCESSED
1260                                    <1>         ;       D      - DIRTY
1261                                    <1>       ;     PAT    - PAGE ATTRIBUTE TABLE INDEX (CACHE BEHAVIOR)
1262                                    <1>       ;     G      - GLOBAL
1263                                    <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
1264                                    <1>       ;
1265                                    <1>       ;
1266                                    <1>       ;; 80386 PAGE TABLE ENTRY (4 KByte Page)
1267                                    <1>       ;  31                                  12 11  9 8 7 6 5 4 3 2 1 0
1268                                    <1>       ; +--------------------------------------+-----+---+-+-+---+-+-+-+
1269                                    <1>             ; |                                        |     | | | | | |U|R| |
1270                                    <1>             ; |     PAGE FRAME BASE ADDRESS 31..12     | AVL |0|0|D|A|0|0|/|/|P|
1271                                    <1>             ; |                                        |     | | | | | |S|W| |
1272                                    <1>             ; +--------------------------------------+-----+---+-+-+---+-+-+-+
1273                                    <1>       ;
1274                                    <1>         ;       P      - PRESENT
1275                                    <1>         ;       R/W    - READ/WRITE
1276                                    <1>         ;       U/S    - USER/SUPERVISOR
1277                                    <1>         ;       D      - DIRTY
1278                                    <1>         ;       AVL    - AVAILABLE FOR SYSTEMS PROGRAMMER USE
1279                                    <1>       ;
1280                                    <1>         ;       NOTE: 0 INDICATES INTEL RESERVED. DO NOT DEFINE.
1281                                    <1>       ;
1282                                    <1>       ;
1283                                    <1>       ;; Invalid Page Table Entry
1284                                    <1>       ; 31                                                        1 0
1285                                    <1>             ; +---------------------------------------------------------+-+
1286                                    <1>             ; |                                                         | |
1287                                    <1>             ; |                          AVAILABLE                      |0|
1288                                    <1>             ; |                                                         | |
1289                                    <1>             ; +---------------------------------------------------------+-+
```

```
1290                                    <1>       ;
1291                                    <1>
1292 00005EB3 53                       <1>       push   ebx
1293 00005EB4 52                       <1>       push   edx
1294 00005EB5 51                       <1>       push   ecx
1295                                    <1>       ;
1296                                    <1>       ; 21/09/2015 (debugging)
1297 00005EB6 FF05[CC030300]           <1>       inc    dword [u.pfcount] ; page fault count for running process
1298 00005EBC FF05[80050300]           <1>       inc    dword [PF_Count] ; total page fault count
1299                                    <1>       ; 28/06/2015
1300                                    <1>       ;mov   edx, [error_code] ; Lower 5 bits are valid
1301 00005EC2 8A15[78050300]           <1>       mov    dl, [error_code]
1302                                    <1>       ;
1303 00005EC8 F6C201                   <1>       test   dl, 1  ; page fault was caused by a non-present page
1304                                    <1>                     ; sign
1305 00005ECB 7422                     <1>       jz     short pfh_alloc_np
1306                                    <1>       ;
1307                                    <1>       ; If it is not a 'write on read only page' type page fault
1308                                    <1>       ; major page fault error with minor reason must be returned without
1309                                    <1>       ; fixing the problem. 'sys_exit with error' will be needed
1310                                    <1>       ; after return here!
1311                                    <1>       ; Page fault will be remedied, by copying page contents
1312                                    <1>       ; to newly allocated page with write permission;
1313                                    <1>       ; sys_fork -> sys_exec -> copy on write, demand paging method is
1314                                    <1>       ; used for working with minimum possible memory usage.
1315                                    <1>       ; sys_fork will duplicate page directory and tables of parent
1316                                    <1>       ; process with 'read only' flag. If the child process attempts to
1317                                    <1>       ; write on these read only pages, page fault will be directed here
1318                                    <1>       ; for allocating a new page with same data/content.
1319                                    <1>       ;
1320                                    <1>       ; IMPORTANT : Retro UNIX 386 v1 (and SINGLIX and TR-DOS)
1321                                    <1>       ; will not force to separate CODE and DATA space
1322                                    <1>       ; in a process/program...
1323                                    <1>       ; CODE segment/section may contain DATA!
1324                                    <1>       ; It is flat, smoth and simplest programming method already as in
1325                                    <1>       ; Retro UNIX 8086 v1 and MS-DOS programs.
1326                                    <1>       ;
1327 00005ECD F6C202                   <1>       test   dl, 2  ; page fault was caused by a page write
1328                                    <1>                     ; sign
1329 00005ED0 0F84AB000000             <1>         jz      pfh_p_err
1330                                    <1>       ; 31/08/2015
1331 00005ED6 F6C204                   <1>       test   dl, 4  ; page fault was caused while CPL = 3 (user mode)
1332                                    <1>                     ; sign.  (U+W+P = 4+2+1 = 7)
1333 00005ED9 0F84A2000000             <1>         jz  pfh_pv_err
1334                                    <1>       ;
1335                                    <1>       ; make a new page and copy the parent's page content
1336                                    <1>       ; as the child's new page content
1337                                    <1>       ;
1338 00005EDF 0F20D3                   <1>       mov    ebx, cr2 ; CR2 contains the linear address
1339                                    <1>                       ; which has caused to page fault
1340 00005EE2 E8A2000000               <1>       call   copy_page
1341 00005EE7 0F828D000000             <1>         jc      pfh_im_err ; insufficient memory
1342                                    <1>       ;
1343 00005EED EB7D                     <1>         jmp     pfh_cpp_ok
1344                                    <1>       ;
1345                                    <1> pfh_alloc_np:
1346 00005EEF E8E7FCFFFF               <1>       call   allocate_page; (allocate a new page)
1347 00005EF4 0F8280000000             <1>         jc      pfh_im_err      ; 'insufficient memory' error
1348                                    <1> pfh_chk_cpl:
1349                                    <1>       ; EAX = Physical (base) address of the allocated (new) page
1350                                    <1>             ; (Lower 12 bits are ZERO, because
1351                                    <1>             ;      the address is on a page boundary)
1352 00005EFA 80E204                   <1>       and    dl, 4  ; CPL = 3 ?
1353 00005EFD 7505                     <1>       jnz    short pfh_um
1354                                    <1>                     ; Page fault handler for kernel/system mode (CPL=0)
1355 00005EFF 0F20DB                   <1>       mov    ebx, cr3 ; CR3 (Control Register 3) contains physical address
1356                                    <1>                       ; of the current/active page directory
1357                                    <1>                       ; (Always kernel/system mode page directory, here!)
1358                                    <1>                       ; Note: Lower 12 bits are 0. (page boundary)
1359 00005F02 EB06                     <1>       jmp    short pfh_get_pde
1360                                    <1>       ;
1361                                    <1> pfh_um:                     ; Page fault handler for user/appl. mode (CPL=3)
1362 00005F04 8B1D[B8030300]           <1>       mov    ebx, [u.pgdir] ; Page directory of current/active process
1363                                    <1>                     ; Physical address of the USER's page directory
1364                                    <1>                     ; Note: Lower 12 bits are 0. (page boundary)
1365                                    <1> pfh_get_pde:
1366 00005F0A 80CA03                   <1>       or     dl, 3  ; USER + WRITE + PRESENT or SYSTEM + WRITE + PRESENT
1367 00005F0D 0F20D1                   <1>       mov    ecx, cr2 ; CR2 contains the virtual address
1368                                    <1>                      ; which has been caused to page fault
1369                                    <1>                      ;
1370 00005F10 C1E914                   <1>       shr    ecx, 20       ; shift 20 bits right
1371 00005F13 80E1FC                   <1>       and    cl, 0FCh ; mask lower 2 bits to get PDE offset
1372                                    <1>       ;
1373 00005F16 01CB                     <1>       add    ebx, ecx ; now, EBX points to the relevant page dir entry
1374 00005F18 8B0B                     <1>       mov    ecx, [ebx] ; physical (base) address of the page table
1375 00005F1A F6C101                   <1>       test   cl, 1   ; check bit 0 is set (1) or not (0).
1376 00005F1D 740B                     <1>       jz     short pfh_set_pde ; Page directory entry is not valid,
1377                                    <1>                              ; set/validate page directory entry
1378 00005F1F 6681E100F0               <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
1379 00005F24 89CB                     <1>       mov    ebx, ecx ; Physical address of the page table
1380 00005F26 89C1                     <1>       mov    ecx, eax ; new page address (physical)
1381 00005F28 EB16                     <1>       jmp    short pfh_get_pte
1382                                    <1> pfh_set_pde:
1383                                    <1>       ;; NOTE: Page directories and page tables never be swapped out!
1384                                    <1>       ;;      (So, we know this PDE is empty or invalid)
1385                                    <1>       ;
1386 00005F2A 08D0                     <1>       or     al, dl  ; lower 3 bits are used as U/S, R/W, P flags
1387 00005F2C 8903                     <1>       mov    [ebx], eax ; Let's put the new page directory entry here !
1388 00005F2E 30C0                     <1>       xor    al, al  ; clear lower (3..8) bits
1389 00005F30 89C3                     <1>       mov    ebx, eax
1390 00005F32 E8A4FCFFFF               <1>       call   allocate_page ; (allocate a new page)
1391 00005F37 7241                     <1>       jc     short pfh_im_err   ; 'insufficient memory' error
1392                                    <1> pfh_spde_1:
1393                                    <1>       ; EAX = Physical (base) address of the allocated (new) page
1394 00005F39 89C1                     <1>       mov    ecx, eax
```

```
1395 00005F3B E815FDFFFF          <1>       call   clear_page ; Clear page content
1396                              <1> pfh_get_pte:
1397 00005F40 0F20D0              <1>       mov    eax, cr2 ; virtual address
1398                              <1>                       ; which has been caused to page fault
1399 00005F43 89C7                <1>       mov    edi, eax ; 20/07/2015
1400 00005F45 C1E80C              <1>       shr    eax, 12       ; shift 12 bit right to get
1401                              <1>                      ; higher 20 bits of the page fault address
1402 00005F48 25FF030000          <1>       and    eax, 3FFh ; mask PDE# bits, the result is PTE# (0 to 1023)
1403 00005F4D C1E002              <1>       shl    eax, 2 ; shift 2 bits left to get PTE offset
1404 00005F50 01C3                <1>       add    ebx, eax ; now, EBX points to the relevant page table entry
1405 00005F52 8B03                <1>       mov    eax, [ebx] ; get previous value of pte
1406                              <1>              ; bit 0 of EAX is always 0 (otherwise we would not be here)
1407 00005F54 21C0                <1>       and    eax, eax
1408 00005F56 7410                <1>       jz     short pfh_gpte_1
1409                              <1>       ; 20/07/2015
1410 00005F58 87D9                <1>       xchg   ebx, ecx ; new page address (physical)
1411 00005F5A 55                  <1>       push   ebp ; 20/07/2015
1412 00005F5B 0F20D5              <1>       mov    ebp, cr2
1413                              <1>              ; ECX = physical address of the page table entry
1414                              <1>              ; EBX = Memory page address (physical!)
1415                              <1>              ; EAX = Swap disk (offset) address
1416                              <1>              ; EBP = virtual address (page fault address)
1417 00005F5E E8B7000000          <1>       call   swap_in
1418 00005F63 5D                  <1>       pop    ebp
1419 00005F64 7210                <1>       jc      short pfh_err_retn
1420 00005F66 87CB                <1>       xchg   ecx, ebx
1421                              <1>              ; EBX = physical address of the page table entry
1422                              <1>              ; ECX = new page
1423                              <1> pfh_gpte_1:
1424 00005F68 08D1                <1>       or     cl, dl ; lower 3 bits are used as U/S, R/W, P flags
1425 00005F6A 890B                <1>       mov    [ebx], ecx ; Let's put the new page table entry here !
1426                              <1> pfh_cpp_ok:
1427                              <1>       ; 20/07/2015
1428 00005F6C 0F20D3              <1>       mov    ebx, cr2
1429 00005F6F E8A6020000          <1>       call   add_to_swap_queue
1430                              <1>       ;
1431                              <1>       ; The new PTE (which contains the new page) will be added to
1432                              <1>       ; the swap queue, here.
1433                              <1>       ; (Later, if memory will become insufficient,
1434                              <1>       ; one page will be swapped out which is at the head of
1435                              <1>       ; the swap queue by using FIFO and access check methods.)
1436                              <1>       ;
1437 00005F74 31C0                <1>       xor    eax, eax  ; 0
1438                              <1>       ;
1439                              <1> pfh_err_retn:
1440 00005F76 59                  <1>       pop    ecx
1441 00005F77 5A                  <1>       pop    edx
1442 00005F78 5B                  <1>       pop    ebx
1443 00005F79 C3                  <1>       retn
1444                              <1>
1445                              <1> pfh_im_err:
1446 00005F7A B8E4000000          <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_IM ; Error code in AX
1447                              <1>                     ; Major (Primary) Error: Page Fault
1448                              <1>                     ; Minor (Secondary) Error: Insufficient Memory !
1449 00005F7F EBF5                <1>       jmp    short pfh_err_retn
1450                              <1>
1451                              <1>
1452                              <1> pfh_p_err: ; 09/03/2015
1453                              <1> pfh_pv_err:
1454                              <1>       ; Page fault was caused by a protection-violation
1455 00005F81 B8E6000000          <1>       mov    eax, ERR_MAJOR_PF + ERR_MINOR_PV ; Error code in AX
1456                              <1>                     ; Major (Primary) Error: Page Fault
1457                              <1>                     ; Minor (Secondary) Error: Protection violation !
1458 00005F86 F9                  <1>       stc
1459 00005F87 EBED                <1>       jmp    short pfh_err_retn
1460                              <1>
1461                              <1> copy_page:
1462                              <1>       ; 22/09/2015
1463                              <1>       ; 21/09/2015
1464                              <1>       ; 19/09/2015
1465                              <1>       ; 07/09/2015
1466                              <1>       ; 31/08/2015
1467                              <1>       ; 20/07/2015
1468                              <1>       ; 05/05/2015
1469                              <1>       ; 03/05/2015
1470                              <1>       ; 18/04/2015
1471                              <1>       ; 12/04/2015
1472                              <1>       ; 30/10/2014
1473                              <1>       ; 18/10/2014 (Retro UNIX 386 v1 - beginning)
1474                              <1>       ;
1475                              <1>       ; INPUT ->
1476                              <1>       ;     EBX = Virtual (linear) address of source page
1477                              <1>       ;          (Page fault address)
1478                              <1>       ; OUTPUT ->
1479                              <1>       ;     EAX = PHYSICAL (real/flat) ADDRESS OF THE ALLOCATED PAGE
1480                              <1>       ;     (corresponding PAGE TABLE ENTRY is mapped/set)
1481                              <1>       ;     EAX = 0 (CF = 1)
1482                              <1>       ;           if there is not a free page to be allocated
1483                              <1>       ;     (page content of the source page will be copied
1484                              <1>       ;     onto the target/new page)
1485                              <1>       ;
1486                              <1>       ; Modified Registers -> ecx, ebx (except EAX)
1487                              <1>       ;
1488 00005F89 56                  <1>       push   esi
1489 00005F8A 57                  <1>       push   edi
1490                              <1>       ;push  ebx
1491                              <1>       ;push  ecx
1492 00005F8B 31F6                <1>       xor    esi, esi
1493 00005F8D C1EB0C              <1>       shr    ebx, 12 ; shift 12 bits right to get PDE & PTE numbers
1494 00005F90 89D9                <1>       mov    ecx, ebx ; save page fault address (as 12 bit shifted)
1495 00005F92 C1EB08              <1>       shr    ebx, 8  ; shift 8 bits right and then
1496 00005F95 80E3FC              <1>       and    bl, 0FCh ; mask lower 2 bits to get PDE offset
1497 00005F98 89DF                <1>       mov    edi, ebx ; save it for the parent of current process
1498 00005F9A 031D[B8030300]      <1>       add    ebx, [u.pgdir] ; EBX points to the relevant page dir entry
1499 00005FA0 8B03                <1>       mov    eax, [ebx] ; physical (base) address of the page table
```

```
1500 00005FA2 662500F0            <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1501 00005FA6 89CB                <1>       mov   ebx, ecx   ; (restore higher 20 bits of page fault address)
1502 00005FA8 81E3FF030000        <1>       and   ebx, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
1503 00005FAE 66C1E302            <1>       shl   bx, 2    ; shift 2 bits left to get PTE offset
1504 00005FB2 01C3                <1>       add   ebx, eax   ; EBX points to the relevant page table entry
1505                              <1>       ; 07/09/2015
1506 00005FB4 66F7030002          <1>         test    word [ebx], PTE_DUPLICATED ; (Does current process share this
1507                              <1>                                ; read only page as a child process?)
1508 00005FB9 7509                <1>       jnz   short cpp_0 ; yes
1509 00005FBB 8B0B                <1>       mov   ecx, [ebx] ; PTE value
1510 00005FBD 6681E100F0          <1>       and   cx, PTE_A_CLEAR ; 0F000h  ; clear page attributes
1511 00005FC2 EB32                <1>       jmp   short cpp_1
1512                              <1> cpp_0:
1513 00005FC4 89FE                <1>       mov   esi, edi
1514 00005FC6 0335[BC030300]      <1>       add   esi, [u.ppgdir] ; the parent's page directory entry
1515 00005FCC 8B06                <1>       mov   eax, [esi] ; physical (base) address of the page table
1516 00005FCE 662500F0            <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1517 00005FD2 89CE                <1>       mov   esi, ecx   ; (restore higher 20 bits of page fault address)
1518 00005FD4 81E6FF030000        <1>       and   esi, 3FFh  ; mask PDE# bits, the result is PTE# (0 to 1023)
1519 00005FDA 66C1E602            <1>       shl   si, 2    ; shift 2 bits left to get PTE offset
1520 00005FDE 01C6                <1>       add   esi, eax   ; EDX points to the relevant page table entry
1521 00005FE0 8B0E                <1>       mov   ecx, [esi] ; PTE value of the parent process
1522                              <1>       ; 21/09/2015
1523 00005FE2 8B03                <1>       mov   eax, [ebx] ; PTE value of the child process
1524 00005FE4 662500F0            <1>       and   ax, PTE_A_CLEAR ; 0F000h ; clear page attributes
1525                              <1>       ;
1526 00005FE8 F6C101              <1>       test  cl, PTE_A_PRESENT ; is it a present/valid page ?
1527 00005FEB 7424                <1>       jz    short cpp_3 ; the parent's page is not same page
1528                              <1>       ;
1529 00005FED 6681E100F0          <1>       and   cx, PTE_A_CLEAR ; 0F000h ; clear page attributes
1530 00005FF2 39C8                <1>       cmp   eax, ecx   ; Same page?
1531 00005FF4 751B                <1>       jne   short cpp_3 ; Parent page and child page are not same
1532                              <1>                         ; Convert child's page to writable page
1533                              <1> cpp_1:
1534 00005FF6 E8E0FBFFFF          <1>       call  allocate_page
1535 00005FFB 721A                <1>       jc    short cpp_4 ; 'insufficient memory' error
1536 00005FFD 21F6                <1>       and   esi, esi    ; check ESI is valid or not
1537 00005FFF 7405                <1>       jz    short cpp_2
1538                              <1>             ; Convert read only page to writable page
1539                              <1>             ;(for the parent of the current process)
1540                              <1>       ;and  word [esi], PTE_A_CLEAR ; 0F000h
1541                              <1>       ; 22/09/2015
1542 00006001 890E                <1>       mov   [esi], ecx
1543 00006003 800E07              <1>       or    byte [esi], PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER
1544                              <1>                           ; 1+2+4 = 7
1545                              <1> cpp_2:
1546 00006006 89C7                <1>       mov   edi, eax ; new page address of the child process
1547                              <1>       ; 07/09/2015
1548 00006008 89CE                <1>       mov   esi, ecx ; the page address of the parent process
1549 0000600A B900040000          <1>       mov   ecx, PAGE_SIZE / 4
1550 0000600F F3A5                <1>       rep   movsd ; 31/08/2015
1551                              <1> cpp_3:
1552 00006011 0C07                <1>       or    al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 1+2+4 = 7
1553 00006013 8903                <1>       mov   [ebx], eax ; Update PTE
1554 00006015 28C0                <1>       sub   al, al ; clear attributes
1555                              <1> cpp_4:
1556                              <1>       ;pop  ecx
1557                              <1>       ;pop  ebx
1558 00006017 5F                  <1>       pop   edi
1559 00006018 5E                  <1>       pop   esi
1560 00006019 C3                  <1>       retn
1561                              <1>
1562                              <1> ;; 28/04/2015
1563                              <1> ;; 24/10/2014
1564                              <1> ;; 21/10/2014 (Retro UNIX 386 v1 - beginning)
1565                              <1> ;; SWAP_PAGE_QUEUE (4096 bytes)
1566                              <1> ;;
1567                              <1> ;;   0000   0001   0002   0003   ....   1020   1021   1022   1023
1568                              <1> ;; +------+------+------+------+-    -+------+------+------+------+
1569                              <1> ;; |  pg1 |  pg2 |  pg3 |  pg4 | .... |pg1021|pg1022|pg1023|pg1024|
1570                              <1> ;; +------+------+------+------+-    -+------+------+------+------+
1571                              <1> ;;
1572                              <1> ;; [swpq_last] = 0 to 4096 (step 4) -> the last position on the queue
1573                              <1> ;;
1574                              <1> ;; Method:
1575                              <1> ;;    Swap page queue is a list of allocated pages with physical
1576                              <1> ;;    addresses (system mode virtual adresses = physical addresses).
1577                              <1> ;;    It is used for 'swap_in' and 'swap_out' procedures.
1578                              <1> ;;    When a new page is being allocated, swap queue is updated
1579                              <1> ;;    by 'swap_queue_shift' procedure, header of the queue (offset 0)
1580                              <1> ;;    is checked for 'accessed' flag. If the 1st page on the queue
1581                              <1> ;;    is 'accessed' or 'read only', it is dropped from the list;
1582                              <1> ;;    other pages from the 2nd to the last (in [swpq_last]) shifted
1583                              <1> ;;    to head then the 2nd page becomes the 1st and '[swpq_last]'
1584                              <1> ;;    offset value becomes it's previous offset value - 4.
1585                              <1> ;;    If the 1st page of the swap page queue is not 'accessed'
1586                              <1> ;;    the queue/list is not shifted.
1587                              <1> ;;    After the queue/list shift, newly allocated page is added
1588                              <1> ;;    to the tail of the queue at the [swpq_count*4] position.
1589                              <1> ;;    But, if [swpq_count] > 1023, the newly allocated page
1590                              <1> ;;    will not be added to the tail of swap page queue.
1591                              <1> ;;
1592                              <1> ;;    During 'swap_out' procedure, swap page queue is checked for
1593                              <1> ;;    the first non-accessed, writable page in the list,
1594                              <1> ;;    from the head to the tail. The list is shifted to left
1595                              <1> ;;    (to the head) till a non-accessed page will be found in the list.
1596                              <1> ;;    Then, this page     is swapped out (to disk) and then it is dropped
1597                              <1> ;;    from the list by a final swap queue shift. [swpq_count] value
1598                              <1> ;;    is changed. If all pages on the queue' are 'accessed',
1599                              <1> ;;    'insufficient memory' error will be returned ('swap_out'
1600                              <1> ;;    procedure will be failed)...
1601                              <1> ;;
1602                              <1> ;;    Note: If the 1st page of the queue is an 'accessed' page,
1603                              <1> ;;    'accessed' flag of the page will be reset (0) and that page
1604                              <1> ;;    (PTE) will be added to the tail of the queue after
```

```
<1> ;;    the check, if [swpq_count] < 1023. If [swpq_count] = 1024
<1> ;;    the queue will be rotated and the PTE in the head will be
<1> ;;    added to the tail after resetting 'accessed' bit.
<1> ;;
<1> ;;
<1> ;;
<1> ;; SWAP DISK/FILE (with 4096 bytes swapped page blocks)
<1> ;;
<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>
0000601A 833D[62050300]00       <1>         cmp     dword [swp_drv], 0
00006021 7646                   <1>       jna   short swpin_dnp_err
                                <1>
00006023 3B05[66050300]         <1>       cmp   eax, [swpd_size]
00006029 734A                   <1>       jnb   short swpin_snp_err
                                <1>
0000602B 56                     <1>       push  esi
0000602C 53                     <1>       push  ebx
0000602D 51                     <1>       push  ecx
0000602E 8B35[62050300]         <1>       mov   esi, [swp_drv]
00006034 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
00006039 50                     <1>       push  eax
0000603A E8AF020000             <1>       call  logical_disk_read
0000603F 58                     <1>       pop   eax
00006040 730C                   <1>       jnc   short swpin_read_ok
                                <1>       ;
00006042 B828000000             <1>       mov   eax, SWP_DISK_READ_ERR ; drive not ready or read error
00006047 A3[C8030300]           <1>       mov   [u.error], eax
0000604C EB17                   <1>       jmp   short swpin_retn
                                <1>       ;
                                <1> swpin_read_ok:
                                <1>       ; EAX = Offset address (logical sector number)
0000604E E80D020000             <1>       call  unlink_swap_block  ; Deallocate swap block
                                <1>       ;
                                <1>       ; EBX = Memory page (buffer) address (physical!)
                                <1>       ; 20/07/2015
00006053 89EB                   <1>       mov   ebx, ebp ; virtual address (page fault address)
00006055 6681E300F0             <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
0000605A 8A1D[B3030300]         <1>       mov   bl, [u.uno] ; current process number
                                <1>       ; EBX = Virtual (Linear) address & process number combination
00006060 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:
00006065 59                     <1>       pop   ecx
00006066 5B                     <1>       pop   ebx
00006067 5E                     <1>       pop   esi
00006068 C3                     <1>       retn
                                <1>
                                <1> swpin_dnp_err:
00006069 B829000000             <1>       mov   eax, SWP_DISK_NOT_PRESENT_ERR
                                <1> swpin_err_retn:
0000606E A3[C8030300]           <1>       mov   [u.error], eax
00006073 F9                     <1>       stc
00006074 C3                     <1>       retn
                                <1>
                                <1> swpin_snp_err:
00006075 B82A000000             <1>       mov   eax, SWP_SECTOR_NOT_PRESENT_ERR
0000607A 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)
```

```
1815                                    <1>       ;
1816                                    <1>       ; INPUT ->
1817                                    <1>       ;     none
1818                                    <1>       ;
1819                                    <1>       ; OUTPUT ->
1820                                    <1>       ;     EAX = Physical page address (which is swapped out
1821                                    <1>       ;           for allocating a new page)
1822                                    <1>       ;     CF = 1 -> swap disk writing error (disk/file not present
1823                                    <1>       ;               or sector not present or drive not ready
1824                                    <1>       ;          EAX = Error code
1825                                    <1>       ;          [u.error] = EAX
1826                                    <1>       ;                  = The last error code for the process
1827                                    <1>       ;                    (will be reset after returning to user)
1828                                    <1>       ;
1829                                    <1>       ; Modified Registers -> none (except EAX)
1830                                    <1>       ;
1831 0000607C 66833D[60050300]01       <1>       cmp    word [swpq_count], 1
1832 00006084 0F82AF000000             <1>         jc      swpout_im_err ; 'insufficient memory'
1833                                    <1>
1834                                    <1>         ;cmp    dword [swp_drv], 1
1835                                    <1>       ;jc    short swpout_dnp_err ; 'swap disk/file not present'
1836                                    <1>
1837 0000608A 833D[6A050300]01         <1>         cmp     dword [swpd_free], 1
1838 00006091 0F828F000000             <1>         jc      swpout_nfspc_err ; 'no free space on swap disk'
1839                                    <1>
1840 00006097 53                       <1>       push   ebx ; *
1841                                    <1> swpout_1:
1842                                    <1>       ; 10/06/2016
1843 00006098 31DB                     <1>       xor    ebx, ebx ; shift the queue and return a PTE value
1844 0000609A E8A1000000               <1>       call   swap_queue_shift
1845 0000609F 21C0                     <1>       and    eax, eax     ; 0 = empty queue (improper entries)
1846 000060A1 0F848A000000             <1>         jz      swpout_npts_err        ; There is not any proper PTE
1847                                    <1>                                      ; pointer in the swap queue
1848                                    <1>       ; EAX = PTE value of the page
1849                                    <1>       ; EBX = PTE address of the page
1850 000060A7 662500F0                 <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
1851                                    <1>       ;
1852                                    <1>       ; 07/06/2016
1853                                    <1>       ; 19/05/2016
1854                                    <1>       ; check this page is in timer events or not
1855                                    <1>
1856                                    <1> swpout_timer_page_0:
1857 000060AB 52                       <1>       push   edx ; **
1858                                    <1>
1859                                    <1>       ; 07/06/2016
1860 000060AC 803D[AB960100]00         <1>       cmp    byte [timer_events], 0
1861 000060B3 762F                     <1>       jna    short swpout_2
1862                                    <1>       ;
1863 000060B5 8A15[AB960100]           <1>       mov    dl, [timer_events]
1864                                    <1>
1865 000060BB 51                       <1>       push   ecx ; ***
1866 000060BC 53                       <1>       push   ebx ; ****
1867 000060BD BB[60040300]             <1>       mov    ebx, timer_set ; beginning address of timer event
1868                                    <1>                             ; structures
1869                                    <1> swpout_timer_page_1:
1870 000060C2 8A0B                     <1>       mov    cl, [ebx]
1871 000060C4 08C9                     <1>       or     cl, cl ; 0 = free, >0 = process number
1872 000060C6 7415                     <1>       jz     short swpout_timer_page_3
1873 000060C8 8B4B0C                   <1>       mov    ecx, [ebx+12] ; response (signal return) address
1874 000060CB 6681E100F0               <1>       and    cx, PTE_A_CLEAR ; clear offset part (right 12 bits)
1875                                    <1>                            ; of the response byte address, to
1876                                    <1>                            ; get beginning of the page address)
1877 000060D0 39C8                     <1>       cmp    eax, ecx
1878 000060D2 7505                     <1>       jne    short swpout_timer_page_2 ; not same page
1879                                    <1>
1880                                    <1>       ; !same page!
1881                                    <1>       ;
1882                                    <1>       ; NOTE: // 19/05/2016 // - TRDOS 386 feature only ! -
1883                                    <1>       ; This page will be used by the kernel to put timer event
1884                                    <1>       ; response (signal return) byte at the requested address;
1885                                    <1>       ; in order to prevent a possible wrong write (while
1886                                    <1>       ; this page is swapped out) on physical memory,
1887                                    <1>       ; we must protect this page against to be swapped out!
1888                                    <1>       ;
1889 000060D4 5B                       <1>       pop    ebx ; ****
1890 000060D5 59                       <1>       pop    ecx ; ***
1891 000060D6 5A                       <1>       pop    edx ; **
1892 000060D7 EBBF                     <1>       jmp    short swpout_1      ; do not swap out this page !
1893                                    <1>
1894                                    <1> swpout_timer_page_2:
1895                                    <1>       ; 07/06/2016
1896 000060D9 FECA                     <1>       dec    dl
1897 000060DB 7405                     <1>       jz     short swpout_timer_page_4
1898                                    <1> swpout_timer_page_3:
1899                                    <1>       ;cmp   ebx, timer_set + 240 ; last timer event (15*16)
1900                                    <1>       ;jnb   short swpout_timer_page_4
1901 000060DD 83C310                   <1>       add    ebx, 16
1902 000060E0 EBE0                     <1>       jmp    short swpout_timer_page_1
1903                                    <1>
1904                                    <1> swpout_timer_page_4:
1905 000060E2 5B                       <1>       pop    ebx ; ****
1906 000060E3 59                       <1>       pop    ecx ; ***
1907                                    <1> swpout_2:
1908 000060E4 89DA                     <1>       mov    edx, ebx            ; Page table entry address
1909 000060E6 89C3                     <1>       mov    ebx, eax            ; Buffer (Page) Address
1910                                    <1>       ;
1911 000060E8 E8A6010000               <1>       call   link_swap_block
1912 000060ED 7304                     <1>       jnc    short swpout_3            ; It may not be needed here
1913                                    <1>                                      ; because [swpd_free] value
1914                                    <1>                                      ; was checked at the beginging.
1915 000060EF 5A                       <1>       pop    edx ; **
1916 000060F0 5B                       <1>       pop    ebx ; *
1917 000060F1 EB33                     <1>       jmp    short swpout_nfspc_err
1918                                    <1> swpout_3:
1919 000060F3 A900000080               <1>       test   eax, 80000000h ; test bit 31 (this may not be needed!)
```

```
1920 000060F8 752C                <1>       jnz    short swpout_nfspc_err  ; 10/06/2016 (bit 31 = 1 !)
1921                              <1>       ;
1922 000060FA 56                  <1>       push   esi ; **
1923 000060FB 51                  <1>       push   ecx ; ***
1924 000060FC 50                  <1>       push   eax ; sector address ; (31 bit !, bit 31 = 0)
1925 000060FD 8B35[62050300]      <1>       mov    esi, [swp_drv]
1926 00006103 B908000000          <1>       mov    ecx, PAGE_SIZE / LOGIC_SECT_SIZE  ; 8 !
1927                              <1>              ; Note: Even if corresponding physical disk's sector
1928                              <1>              ; size different than 512 bytes, logical disk sector
1929                              <1>              ; size is 512 bytes and disk writing procedure
1930                              <1>              ; will be performed for writing 4096 bytes
1931                              <1>              ; (2*2048, 8*512).
1932                              <1>       ; ESI = Logical disk description table address
1933                              <1>       ; EBX = Buffer (Page) address
1934                              <1>       ; EAX = Sector adress (offset address, logical sector number)
1935                              <1>       ; ECX = Sector count ; 8 sectors
1936                              <1>       ; edx = PTE address
1937 00006108 E8E2010000          <1>       call   logical_disk_write
1938                              <1>       ; edx = PTE address
1939 0000610D 59                  <1>       pop    ecx ; sector address
1940 0000610E 730C                <1>       jnc    short swpout_write_ok
1941                              <1>       ;
1942                              <1>       ;; call      unlink_swap_block ; this block must be left as 'in use'
1943                              <1> swpout_dw_err:
1944 00006110 B82C000000          <1>       mov    eax, SWP_DISK_WRITE_ERR ; drive not ready or write error
1945 00006115 A3[C8030300]        <1>       mov    [u.error], eax
1946 0000611A EB06                <1>       jmp    short swpout_retn
1947                              <1>       ;
1948                              <1> swpout_write_ok:
1949                              <1>       ; EBX = Buffer (page) address
1950                              <1>       ; EDX = Page Table Entry address
1951                              <1>       ; ECX = Swap disk sector (file block) address (31 bit)
1952 0000611C D1E1                <1>       shl    ecx, 1  ; 31 bit sector address from bit 1 to bit 31
1953 0000611E 890A                <1>       mov    [edx], ecx
1954                              <1>              ; bit 0 = 0 (swapped page)
1955 00006120 89D8                <1>       mov    eax, ebx
1956                              <1> swpout_retn:
1957 00006122 59                  <1>       pop    ecx ; ***
1958 00006123 5E                  <1>       pop    esi ; **
1959 00006124 5B                  <1>       pop    ebx ; *
1960 00006125 C3                  <1>       retn
1961                              <1>
1962                              <1> ;swpout_dnp_err:
1963                              <1> ;     mov    eax, SWP_DISK_NOT_PRESENT_ERR ; disk not present
1964                              <1> ;     jmp    short swpout_err_retn
1965                              <1> swpout_nfspc_err:
1966 00006126 B82B000000          <1>       mov    eax, SWP_NO_FREE_SPACE_ERR ; no free space
1967                              <1> swpout_err_retn:
1968 0000612B A3[C8030300]        <1>       mov    [u.error], eax
1969                              <1>       ;stc
1970 00006130 C3                  <1>       retn
1971                              <1> swpout_npts_err:
1972 00006131 B82D000000          <1>       mov    eax, SWP_NO_PAGE_TO_SWAP_ERR
1973 00006136 5B                  <1>       pop    ebx
1974 00006137 EBF2                <1>       jmp    short swpout_err_retn
1975                              <1> swpout_im_err:
1976 00006139 B804000000          <1>       mov    eax, ERR_MINOR_IM ; insufficient (out of) memory
1977 0000613E EBEB                <1>       jmp    short swpout_err_retn
1978                              <1>
1979                              <1> swap_queue_shift:
1980                              <1>       ; 26/03/2017
1981                              <1>       ; 10/06/2016
1982                              <1>       ; 09/06/2016 - TRDOS 386 (TRDOS v2.0)
1983                              <1>       ; 23/10/2014 - 20/07/2015 (Retro UNIX 386 v1)
1984                              <1>       ;
1985                              <1>       ; INPUT ->
1986                              <1>       ;     EBX = Virtual (linear) address (bit 12 to 31)
1987                              <1>       ;           and process number combination (bit 0 to 11)
1988                              <1>       ;     EBX = 0 -> shift/drop from the head (offset 0)
1989                              <1>       ;
1990                              <1>       ; OUTPUT ->
1991                              <1>       ;     If EBX input > 0
1992                              <1>       ;        the queue will be shifted 4 bytes (dword),
1993                              <1>       ;        from the tail to the head, up to entry offset
1994                              <1>       ;        which points to EBX input value or nothing
1995                              <1>       ;        to do if EBX value is not found on the queue.
1996                              <1>       ;        (The entry -with EBX value- will be removed
1997                              <1>       ;        from the queue if it is found.)
1998                              <1>       ;
1999                              <1>       ;        EAX = 0
2000                              <1>       ;
2001                              <1>       ;     If EBX input = 0
2002                              <1>       ;        the queue will be shifted 4 bytes (dword),
2003                              <1>       ;        from the tail to the head, if the PTE address
2004                              <1>       ;        which is pointed in head of the queue is marked
2005                              <1>       ;        as "accessed" or it is marked as "non present".
2006                              <1>       ;        (If "accessed" flag of the PTE -which is pointed
2007                              <1>       ;        in the head- is set -to 1-, it will be reset
2008                              <1>       ;        -to 0- and then, the queue will be rotated
2009                              <1>       ;        -without dropping pointer of the PTE from
2010                              <1>       ;        the queue- for 4 bytes on head to tail direction.
2011                              <1>       ;        Pointer in the head will be moved into the tail,
2012                              <1>       ;        other PTEs will be shifted on head direction.)
2013                              <1>       ;
2014                              <1>       ;        Swap queue will be shifted up to the first
2015                              <1>       ;        'present' or 'non accessed' page will be found
2016                              <1>       ;        (as pointed) on the queue head (then it will be
2017                              <1>         ;          removed/dropped from the queue).
2018                              <1>       ;
2019                              <1>       ;        EAX (> 0) = PTE value of the page which is
2020                              <1>       ;             (it's pointer -virtual address-) dropped
2021                              <1>       ;             (removed) from swap queue.
2022                              <1>       ;        EBX = PTE address of the page (if EAX > 0)
2023                              <1>       ;              which is (it's pointer -virtual address-)
2024                              <1>       ;             dropped (removed) from swap queue.
```

```
2025                                    <1>       ;
2026                                    <1>       ;         EAX = 0 -> empty swap queue !
2027                                    <1>       ;
2028                                    <1>       ; Modified Registers -> EAX, EBX
2029                                    <1>       ;
2030 00006140 0FB705[60050300]          <1>       movzx   eax, word [swpq_count]  ; Max. 1024
2031 00006147 6621C0                    <1>       and    ax, ax
2032 0000614A 7431                      <1>       jz     short swpqs_retn
2033 0000614C 57                        <1>       push   edi
2034 0000614D 56                        <1>       push   esi
2035 0000614E 51                        <1>       push   ecx
2036 0000614F BE00E00800                <1>       mov    esi, swap_queue
2037 00006154 89C1                      <1>       mov    ecx, eax
2038 00006156 09DB                      <1>       or     ebx, ebx
2039 00006158 7424                      <1>       jz     short swpqs_7
2040                                    <1> swpqs_1:
2041 0000615A AD                        <1>       lodsd
2042 0000615B 39D8                      <1>       cmp    eax, ebx
2043 0000615D 7406                      <1>       je     short swpqs_2
2044 0000615F E2F9                      <1>       loop   swpqs_1
2045                                    <1>       ; 10/06/2016
2046 00006161 29C0                      <1>       sub    eax, eax
2047 00006163 EB15                      <1>       jmp    short swpqs_6
2048                                    <1> swpqs_2:
2049 00006165 89F7                      <1>       mov    edi, esi
2050 00006167 83EF04                    <1>       sub    edi, 4
2051                                    <1> swpqs_3:
2052 0000616A 66FF0D[60050300]          <1>       dec    word [swpq_count]
2053 00006171 7403                      <1>       jz     short swpqs_5
2054                                    <1> swpqs_4:
2055 00006173 49                        <1>       dec    ecx
2056 00006174 F3A5                      <1>       rep    movsd  ; shift up (to the head)
2057                                    <1> swpqs_5:
2058 00006176 31C0                      <1>       xor    eax, eax
2059 00006178 8907                      <1>       mov    [edi], eax
2060                                    <1> swpqs_6:
2061 0000617A 59                        <1>       pop    ecx
2062 0000617B 5E                        <1>       pop    esi
2063 0000617C 5F                        <1>       pop    edi
2064                                    <1> swpqs_retn:
2065 0000617D C3                        <1>       retn
2066                                    <1> swpqs_7:
2067 0000617E 89F7                      <1>       mov    edi, esi ; head
2068 00006180 AD                        <1>       lodsd
2069                                    <1>       ; 20/07/2015
2070 00006181 89C3                      <1>       mov    ebx, eax
2071 00006183 81E300F0FFFF              <1>       and    ebx, ~PAGE_OFF ; ~0FFFh
2072                                    <1>                     ; ebx = virtual address (at page boundary)
2073 00006189 25FF0F0000                <1>       and    eax, PAGE_OFF ; 0FFFh
2074                                    <1>                     ; ax = process number (1 to 4095)
2075 0000618E 3A05[B3030300]            <1>       cmp    al, [u.uno]
2076                                    <1>                     ; Max. 16 (nproc) processes for Retro UNIX 386 v1
2077 00006194 7507                      <1>       jne    short swpqs_8
2078 00006196 A1[B8030300]              <1>       mov    eax, [u.pgdir]
2079 0000619B EB28                      <1>       jmp    short swpqs_9
2080                                    <1> swpqs_8:
2081                                    <1>       ; 09/06/2016
2082 0000619D 80B8[AF000300]00          <1>       cmp    byte [eax+p.stat-1], 0
2083 000061A4 76C4                      <1>       jna    short swpqs_3     ; free (or terminated) process
2084 000061A6 80B8[AF000300]02          <1>       cmp    byte [eax+p.stat-1], 2 ; waiting
2085 000061AD 77BB                      <1>       ja     short swpqs_3         ; zombie (3) or undefined ?
2086                                    <1>
2087                                    <1>       ;shl   ax, 2
2088 000061AF C0E002                    <1>       shl    al, 2
2089 000061B2 8B80[BC000300]            <1>       mov    eax, [eax+p.upage-4]
2090 000061B8 09C0                      <1>       or     eax, eax
2091 000061BA 74AE                      <1>       jz     short swpqs_3 ; invalid upage
2092 000061BC 83C05C                    <1>       add    eax, u.pgdir - user
2093                                    <1>                      ; u.pgdir value for the process
2094                                    <1>                      ; is in [eax]
2095 000061BF 8B00                      <1>       mov    eax, [eax]
2096 000061C1 21C0                      <1>       and    eax, eax
2097 000061C3 74A5                      <1>       jz     short swpqs_3 ; invalid page directory
2098                                    <1> swpqs_9:
2099 000061C5 52                        <1>       push   edx
2100                                    <1>       ; eax = page directory
2101                                    <1>       ; ebx = virtual address
2102 000061C6 E82BFBFFFF                <1>       call   get_pte
2103 000061CB 89D3                      <1>       mov    ebx, edx     ; PTE address
2104 000061CD 5A                        <1>       pop    edx
2105                                    <1>       ; 10/06/2016
2106 000061CE 723A                      <1>       jc     short swpqs_13 ; empty PDE
2107                                    <1>       ; EAX = PTE value
2108 000061D0 A801                      <1>       test   al, PTE_A_PRESENT ; bit 0 = 1
2109 000061D2 7436                      <1>       jz     short swpqs_13  ; Drop non-present page
2110                                    <1>                            ; from the queue (head)
2111 000061D4 A802                      <1>       test   al, PTE_A_WRITE     ; bit 1 = 0 (read only)
2112 000061D6 7432                      <1>       jz     short swpqs_13  ; Drop read only page
2113                                    <1>                            ; from the queue (head)
2114                                    <1>       ;test  al, PTE_A_ACCESS ; bit 5 = 1 (Accessed)
2115                                    <1>       ;jnz   short swpqs_11  ; present
2116                                    <1>                            ; accessed page
2117 000061D8 0FBAF005                  <1>         btr     eax, PTE_A_ACCESS_BIT ; reset 'accessed' bit
2118 000061DC 7210                      <1>       jc     short swpqs_11  ; accessed page
2119                                    <1>
2120 000061DE 49                        <1>       dec    ecx
2121 000061DF 66890D[60050300]          <1>       mov    [swpq_count], cx
2122 000061E6 7402                      <1>         jz      short swpqs_10
2123                                    <1>             ; esi = head + 4
2124                                    <1>             ; edi = head
2125 000061E8 F3A5                      <1>       rep    movsd   ; n = 1 to k-1, [n - 1] = [n]
2126                                    <1> swpqs_10:
2127 000061EA 890F                      <1>       mov    [edi], ecx ; 0
2128 000061EC EB8C                      <1>       jmp    short swpqs_6 ; 26/03/2017
2129                                    <1>
```

```
2130                                    <1> swpqs_11:
2131 000061EE 8903                     <1>         mov    [ebx], eax     ; save changed attribute
2132                                    <1>         ; Rotation (head -> tail)
2133 000061F0 49                       <1>         dec    ecx     ; entry count -> last entry number
2134 000061F1 74F7                     <1>         jz     short swpqs_10
2135                                    <1>                ; esi = head + 4
2136                                    <1>                ; edi = head
2137 000061F3 8B07                     <1>         mov    eax, [edi] ; 20/07/2015
2138 000061F5 F3A5                     <1>         rep    movsd   ; n = 1 to k-1, [n - 1] = [n]
2139 000061F7 8907                     <1>         mov    [edi], eax ; head -> tail ; [k] = [1]
2140                                    <1>
2141 000061F9 668B0D[60050300]         <1>         mov    cx, [swpq_count]
2142                                    <1>
2143                                    <1> swpqs_12:
2144 00006200 BE00E00800               <1>         mov    esi, swap_queue ; head
2145 00006205 E974FFFFFF               <1>           jmp     swpqs_7
2146                                    <1>
2147                                    <1> swpqs_13:
2148 0000620A 49                       <1>         dec    ecx
2149 0000620B 66890D[60050300]         <1>         mov    [swpq_count], cx
2150 00006212 0F845EFFFFFF             <1>           jz      swpqs_5
2151 00006218 EBE6                     <1>         jmp    short swpqs_12
2152                                    <1>
2153                                    <1> add_to_swap_queue:
2154                                    <1> ; temporary - 16/09/2015
2155 0000621A C3                       <1> retn
2156                                    <1>         ; 20/02/2017
2157                                    <1>         ; 20/07/2015
2158                                    <1>         ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
2159                                    <1>         ;
2160                                    <1>         ; Adds new page to swap queue
2161                                    <1>         ; (page directories and page tables must not be added
2162                                    <1>         ; to swap queue)
2163                                    <1>         ;
2164                                    <1>         ; INPUT ->
2165                                    <1>         ;      EBX = Linear (Virtual) addr for current process
2166                                    <1>         ;      [u.uno]
2167                                    <1>         ;      20/02/2017
2168                                    <1>         ;      (Linear address = CORE + user's virtual address)
2169                                    <1>         ;
2170                                    <1>         ; OUTPUT ->
2171                                    <1>         ;      EAX = [swpq_count]
2172                                    <1>         ;            (after the PTE has been added)
2173                                    <1>         ;      EAX = 0 -> Swap queue is full, (1024 entries)
2174                                    <1>         ;            the PTE could not be added.
2175                                    <1>         ;
2176                                    <1>         ; Modified Registers -> EAX
2177                                    <1>         ;
2178 0000621B 53                       <1>         push   ebx
2179 0000621C 6681E300F0               <1>           and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
2180 00006221 8A1D[B3030300]           <1>         mov    bl, [u.uno] ; current process number
2181 00006227 E814FFFFFF               <1>         call   swap_queue_shift ; drop from the queue if
2182                                    <1>                                 ; it is already on the queue
2183                                    <1>                ; then add it to the tail of the queue
2184 0000622C 0FB705[60050300]         <1>         movzx  eax, word [swpq_count]
2185 00006233 663D0004                 <1>         cmp    ax, 1024
2186 00006237 7205                     <1>         jb     short atsq_1
2187 00006239 6629C0                   <1>         sub    ax, ax
2188 0000623C 5B                       <1>         pop    ebx
2189 0000623D C3                       <1>         retn
2190                                    <1> atsq_1:
2191 0000623E 56                       <1>         push   esi
2192 0000623F BE00E00800               <1>         mov    esi, swap_queue
2193 00006244 6621C0                   <1>         and    ax, ax
2194 00006247 740A                     <1>         jz     short atsq_2
2195 00006249 66C1E002                 <1>         shl    ax, 2  ; convert to offset
2196 0000624D 01C6                     <1>         add    esi, eax
2197 0000624F 66C1E802                 <1>         shr    ax, 2
2198                                    <1> atsq_2:
2199 00006253 6640                     <1>         inc    ax
2200 00006255 891E                     <1>         mov    [esi], ebx ; Virtual address + [u.uno] combination
2201 00006257 66A3[60050300]           <1>         mov    [swpq_count], ax
2202 0000625D 5E                       <1>         pop    esi
2203 0000625E 5B                       <1>         pop    ebx
2204 0000625F C3                       <1>         retn
2205                                    <1>
2206                                    <1> unlink_swap_block:
2207                                    <1>         ; 15/09/2015
2208                                    <1>         ; 30/04/2015
2209                                    <1>         ; 18/04/2015
2210                                    <1>         ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
2211                                    <1>         ;
2212                                    <1>         ; INPUT ->
2213                                    <1>         ;      EAX = swap disk/file offset address
2214                                    <1>         ;            (bit 1 to bit 31)
2215                                    <1>         ; OUTPUT ->
2216                                    <1>         ;      [swpd_free] is increased
2217                                    <1>         ;      (corresponding SWAP DISK ALLOC. TABLE bit is SET)
2218                                    <1>         ;
2219                                    <1>         ; Modified Registers -> EAX
2220                                    <1>         ;
2221 00006260 53                       <1>         push   ebx
2222 00006261 52                       <1>         push   edx
2223                                    <1>         ;
2224 00006262 C1E804                   <1>         shr    eax, SECTOR_SHIFT+1  ;3+1 ; shift sector address to
2225                                    <1>                                   ; 3 bits right
2226                                    <1>                                   ; to get swap block/page number
2227 00006265 89C2                     <1>         mov    edx, eax
2228                                    <1>         ; 15/09/2015
2229 00006267 C1EA03                   <1>         shr    edx, 3            ; to get offset to S.A.T.
2230                                    <1>                                   ; (1 allocation bit = 1 page)
2231                                    <1>                                   ; (1 allocation bytes = 8 pages)
2232 0000626A 80E2FC                   <1>         and    dl, 0FCh          ; clear lower 2 bits
2233                                    <1>                                   ; (to get 32 bit position)
2234                                    <1>         ;
```

```
2235 0000626D BB00000D00          <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table address
2236 00006272 01D3                <1>       add    ebx, edx
2237 00006274 83E01F              <1>       and    eax, 1Fh          ; lower 5 bits only
2238                              <1>                                 ; (allocation bit position)
2239 00006277 3B05[6E050300]      <1>       cmp    eax, [swpd_next]     ; is the new free block addr. lower
2240                              <1>                                 ; than the address in 'swpd_next' ?
2241                              <1>                                 ; (next/first free block value)
2242 0000627D 7305                <1>       jnb    short uswpbl_1          ; no
2243 0000627F A3[6E050300]        <1>       mov    [swpd_next], eax     ; yes
2244                              <1> uswpbl_1:
2245 00006284 0FAB03              <1>       bts    [ebx], eax        ; unlink/release/deallocate block
2246                              <1>                                 ; set relevant bit to 1.
2247                              <1>                                 ; set CF to the previous bit value
2248 00006287 F5                  <1>       cmc                       ; complement carry flag
2249 00006288 7206                <1>       jc     short uswpbl_2          ; do not increase swfd_free count
2250                              <1>                                 ; if the block is already deallocated
2251                              <1>                                 ; before.
2252 0000628A FF05[6A050300]      <1>         inc     dword [swpd_free]
2253                              <1> uswpbl_2:
2254 00006290 5A                  <1>       pop    edx
2255 00006291 5B                  <1>       pop    ebx
2256 00006292 C3                  <1>       retn
2257                              <1>
2258                              <1> link_swap_block:
2259                              <1>       ; 01/07/2015
2260                              <1>       ; 18/04/2015
2261                              <1>       ; 24/10/2014 (Retro UNIX 386 v1 - beginning)
2262                              <1>       ;
2263                              <1>       ; INPUT -> none
2264                              <1>       ;
2265                              <1>       ; OUTPUT ->
2266                              <1>       ;      EAX = OFFSET ADDRESS OF THE ALLOCATED BLOCK (4096 bytes)
2267                              <1>       ;            in sectors (corresponding
2268                              <1>       ;            SWAP DISK ALLOCATION TABLE bit is RESET)
2269                              <1>       ;
2270                              <1>       ;      CF = 1 and EAX = 0
2271                              <1>       ;               if there is not a free block to be allocated
2272                              <1>       ;
2273                              <1>       ; Modified Registers -> none (except EAX)
2274                              <1>       ;
2275                              <1>
2276                              <1>       ;mov   eax, [swpd_free]
2277                              <1>       ;and   eax, eax
2278                              <1>       ;jz    short out_of_swpspc
2279                              <1>       ;
2280 00006293 53                  <1>       push   ebx
2281 00006294 51                  <1>       push   ecx
2282                              <1>       ;
2283 00006295 BB00000D00          <1>       mov    ebx, swap_alloc_table ; Swap Allocation Table offset
2284 0000629A 89D9                <1>       mov    ecx, ebx
2285 0000629C 031D[6E050300]      <1>       add    ebx, [swpd_next] ; Free block searching starts from here
2286                              <1>                               ; next_free_swap_block >> 5
2287 000062A2 030D[72050300]      <1>       add    ecx, [swpd_last] ; Free block searching ends here
2288                              <1>                               ; (total_swap_blocks - 1) >> 5
2289                              <1> lswbl_scan:
2290 000062A8 39CB                <1>       cmp    ebx, ecx
2291 000062AA 770A                <1>       ja     short lswbl_notfound
2292                              <1>       ;
2293 000062AC 0FBC03              <1>       bsf    eax, [ebx] ; Scans source operand for first bit set (1).
2294                              <1>                         ; Clears ZF if a bit is found set (1) and
2295                              <1>                         ; loads the destination with an index to
2296                              <1>                         ; first set bit. (0 -> 31)
2297                              <1>                         ; Sets ZF to 1 if no bits are found set.
2298                              <1>       ; 01/07/2015
2299 000062AF 751C                <1>       jnz    short lswbl_found ; ZF = 0 -> a free block has been found
2300                              <1>                        ;
2301                              <1>                        ; NOTE:  a Swap Disk Allocation Table bit
2302                              <1>                        ;       with value of 1 means
2303                              <1>                        ;       the corresponding page is free
2304                              <1>                        ;       (Retro UNIX 386 v1 feaure only!)
2305 000062B1 83C304              <1>       add    ebx, 4
2306                              <1>                        ; We return back for searching next page block
2307                              <1>                        ; NOTE: [swpd_free] is not ZERO; so,
2308                              <1>                        ;      we always will find at least 1 free block here.
2309 000062B4 EBF2                <1>       jmp          short lswbl_scan
2310                              <1>       ;
2311                              <1> lswbl_notfound:
2312 000062B6 81E900000D00        <1>       sub    ecx, swap_alloc_table
2313 000062BC 890D[6E050300]      <1>       mov    [swpd_next], ecx ; next/first free page = last page
2314                              <1>                               ; (unlink_swap_block procedure will change it)
2315 000062C2 31C0                <1>       xor    eax, eax
2316 000062C4 A3[6A050300]        <1>       mov    [swpd_free], eax
2317 000062C9 F9                  <1>       stc
2318                              <1> lswbl_ok:
2319 000062CA 59                  <1>       pop    ecx
2320 000062CB 5B                  <1>       pop    ebx
2321 000062CC C3                  <1>       retn
2322                              <1>       ;
2323                              <1> ;out_of_swpspc:
2324                              <1> ;     stc
2325                              <1> ;     retn
2326                              <1>
2327                              <1> lswbl_found:
2328 000062CD 89D9                <1>       mov    ecx, ebx
2329 000062CF 81E900000D00        <1>       sub    ecx, swap_alloc_table
2330 000062D5 890D[6E050300]      <1>       mov    [swpd_next], ecx ; Set first free block searching start
2331                              <1>                               ; address/offset (to the next)
2332 000062DB FF0D[6A050300]      <1>         dec     dword [swpd_free] ; 1 block has been allocated (X = X-1)
2333                              <1>       ;
2334 000062E1 0FB303              <1>       btr    [ebx], eax    ; The destination bit indexed by the source value
2335                              <1>                            ; is copied into the Carry Flag and then cleared
2336                              <1>                            ; in the destination.
2337                              <1>                            ;
2338                              <1>                            ; Reset the bit which is corresponding to the
2339                              <1>                            ; (just) allocated block.
```

```
2340 000062E4 C1E105              <1>       shl    ecx, 5        ; (block offset * 32) + block index
2341 000062E7 01C8                <1>       add    eax, ecx      ; = block number
2342 000062E9 C1E003              <1>       shl    eax, SECTOR_SHIFT ; 3, sector (offset) address of the block
2343                              <1>                             ; 1 block =  8 sectors
2344                              <1>       ;
2345                              <1>       ; EAX = offset address of swap disk/file sector (beginning of the block)
2346                              <1>       ;
2347                              <1>       ; NOTE: The relevant page table entry will be updated
2348                              <1>       ;       according to this EAX value...
2349                              <1>       ;
2350 000062EC EBDC                <1>       jmp    short lswbl_ok
2351                              <1>
2352                              <1> logical_disk_read:
2353                              <1>       ; 20/07/2015
2354                              <1>       ; 09/03/2015 (temporary code here)
2355                              <1>       ;
2356                              <1>       ; INPUT ->
2357                              <1>       ;      ESI = Logical disk description table address
2358                              <1>       ;      EBX = Memory page (buffer) address (physical!)
2359                              <1>       ;      EAX = Sector adress (offset address, logical sector number)
2360                              <1>       ;      ECX = Sector count
2361                              <1>       ;
2362                              <1>       ;
2363 000062EE C3                  <1>       retn
2364                              <1>
2365                              <1> logical_disk_write:
2366                              <1>       ; 20/07/2015
2367                              <1>       ; 09/03/2015 (temporary code here)
2368                              <1>       ;
2369                              <1>       ; INPUT ->
2370                              <1>       ;      ESI = Logical disk description table address
2371                              <1>       ;      EBX = Memory page (buffer) address (physical!)
2372                              <1>       ;      EAX = Sector adress (offset address, logical sector number)
2373                              <1>       ;      ECX = Sector count
2374                              <1>       ;
2375 000062EF C3                  <1>       retn
2376                              <1>
2377                              <1> get_physical_addr:
2378                              <1>       ; 26/03/2017
2379                              <1>       ; 20/02/2017
2380                              <1>       ; 27/05/2016 - TRDOS 386 (TRDOS v2.0)
2381                              <1>       ; 18/10/2015
2382                              <1>       ; 29/07/2015
2383                              <1>       ; 20/07/2015
2384                              <1>       ; 04/06/2015
2385                              <1>       ; 20/05/2015
2386                              <1>       ; 28/04/2015
2387                              <1>       ; 18/04/2015
2388                              <1>       ; Get physical address
2389                              <1>       ;     (allocates a new page for user if it is not present)
2390                              <1>       ;
2391                              <1>       ; (This subroutine is needed for mapping user's virtual
2392                              <1>       ; (buffer) address to physical address (of the buffer).)
2393                              <1>       ; ('sys write', 'sys read' system calls...)
2394                              <1>       ;
2395                              <1>       ; INPUT ->
2396                              <1>       ;      EBX = virtual address
2397                              <1>       ;      u.pgdir = page directory (physical) address
2398                              <1>       ;
2399                              <1>       ; OUTPUT ->
2400                              <1>       ;      EAX = physical address
2401                              <1>       ;      EBX = linear address
2402                              <1>       ;      EDX = physical address of the page frame
2403                              <1>       ;            (with attribute bits)
2404                              <1>       ;      ECX = byte count within the page frame
2405                              <1>       ;
2406                              <1>       ; Modified Registers -> EAX, EBX, ECX, EDX
2407                              <1>       ;
2408 000062F0 81C300004000        <1>       add    ebx, CORE ; 18/10/2015
2409                              <1> get_physical_addr_x: ; 27/05/2016
2410 000062F6 A1[B8030300]        <1>       mov    eax, [u.pgdir]
2411 000062FB E8F6F9FFFF          <1>       call   get_pte
2412                              <1>             ; EDX = Page table entry address (if CF=0)
2413                              <1>               ;       Page directory entry address (if CF=1)
2414                              <1>             ;       (Bit 0 value is 0 if PT is not present)
2415                              <1>             ; EAX = Page table entry value (page address)
2416                              <1>             ;     CF = 1 -> PDE not present or invalid ?
2417 00006300 731C                <1>       jnc    short gpa_1
2418                              <1>       ;
2419 00006302 E8D4F8FFFF          <1>       call   allocate_page
2420 00006307 7248                <1>       jc     short gpa_im_err  ; 'insufficient memory' error
2421                              <1> gpa_0:
2422 00006309 E847F9FFFF          <1>       call   clear_page
2423                              <1>       ; EAX = Physical (base) address of the allocated (new) page
2424 0000630E 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER ; 4+2+1 = 7
2425                              <1>                       ; lower 3 bits are used as U/S, R/W, P flags
2426                              <1>                       ; (user, writable, present page)
2427 00006310 8902                <1>       mov    [edx], eax ; Let's put the new page directory entry here !
2428 00006312 A1[B8030300]        <1>       mov    eax, [u.pgdir]
2429 00006317 E8DAF9FFFF          <1>       call   get_pte
2430 0000631C 7233                <1>       jc     short gpa_im_err ; 'insufficient memory' error
2431                              <1> gpa_1:
2432                              <1>       ; EAX = PTE value, EDX = PTE address
2433 0000631E A801                <1>       test   al, PTE_A_PRESENT
2434 00006320 751F                <1>       jnz    short gpa_3 ; 26/03/2017
2435 00006322 09C0                <1>       or     eax, eax
2436 00006324 7456                <1>       jz     short gpa_7  ; Allocate a new page
2437                              <1>       ; 20/07/2015
2438 00006326 55                  <1>       push   ebp
2439 00006327 89DD                <1>       mov    ebp, ebx ; virtual (linear) address
2440                              <1>       ; reload swapped page
2441 00006329 E878000000          <1>       call   reload_page ; 28/04/2015
2442 0000632E 5D                  <1>       pop    ebp
2443 0000632F 724A                <1>       jc     short gpa_retn
2444                              <1> gpa_2:
```

```
2445                            <1>       ; 26/03/2017
2446                            <1>       ; 20/02/2017
2447                            <1>       ; If a page will contain a Signal Response Byte
2448                            <1>       ; it must not be swapped out, because
2449                            <1>       ; timer service or irq callback service
2450                            <1>       ; will write a signal return/response byte
2451                            <1>       ; directly by using physical address of Signal
2452                            <1>       ; Response Byte.(Even if process is not running,
2453                            <1>       ; or it is running with swapped out pages.)
2454                            <1>       ;
2455                            <1>       ; 'no_page_swap' will be set by 'systimer' or
2456                            <1>       ; 'syscalbac' sistem functions/calls. (*)
2457                            <1>       ;
2458 00006331 803D[EA9B0100]00    <1>       cmp    byte [no_page_swap], 0
2459 00006338 761D              <1>       jna    short gpa_4 ; this page can be swapped out
2460                            <1>       ; this page must not be swapped out
2461                            <1>       ; but 'no_page_swap' must be reset here
2462                            <1>       ; imediately for other callers (*)
2463                            <1>       ; (otherwise, swap queue would not be long enough)
2464 0000633A E84B000000          <1>       call   gpa_8 ; 26/03/2017
2465 0000633F EB1D              <1>       jmp    short gpa_5
2466                            <1> gpa_3:
2467                            <1>       ; 26/03/2017
2468 00006341 803D[EA9B0100]00    <1>       cmp    byte [no_page_swap], 0
2469 00006348 7618              <1>       jna    short gpa_6 ; this page can be swapped out
2470 0000634A E83B000000          <1>       call   gpa_8
2471 0000634F EB11              <1>       jmp    short gpa_6
2472                            <1>
2473                            <1> gpa_im_err:
2474 00006351 B804000000          <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
2475                            <1>                                  ; Major error = 0 (No protection fault)
2476 00006356 C3                <1>       retn
2477                            <1> gpa_4:
2478                            <1>       ; 20/07/2015
2479                            <1>       ; 20/05/2015
2480                            <1>       ; add this page to swap queue
2481 00006357 50                <1>       push   eax
2482                            <1>       ; EBX = Linear (CORE+virtual) address ; 20/02/2017
2483 00006358 E8BDFEFFFF          <1>       call   add_to_swap_queue
2484 0000635D 58                <1>       pop    eax
2485                            <1> gpa_5:
2486                            <1>              ; PTE address in EDX
2487                            <1>              ; virtual address in EBX
2488                            <1>       ; EAX = memory page address
2489 0000635E 0C07              <1>       or     al, PTE_A_PRESENT + PTE_A_USER + PTE_A_WRITE
2490                            <1>                              ; present flag, bit 0 = 1
2491                            <1>                              ; user flag, bit 2 = 1
2492                            <1>                              ; writable flag, bit 1 = 1
2493 00006360 8902              <1>       mov    [edx], eax  ; Update PTE value
2494                            <1> gpa_6:
2495                            <1>       ; 18/10/2015
2496 00006362 89D9              <1>       mov    ecx, ebx
2497 00006364 81E1FF0F0000        <1>       and    ecx, PAGE_OFF
2498 0000636A 89C2              <1>       mov    edx, eax
2499 0000636C 662500F0          <1>       and    ax, PTE_A_CLEAR
2500 00006370 01C8              <1>       add    eax, ecx
2501 00006372 F7D9              <1>       neg    ecx ; 1 -> -1 (0FFFFFFFFh), 4095 (0FFFh) -> -4095
2502 00006374 81C100100000        <1>       add    ecx, PAGE_SIZE
2503 0000637A F8                <1>       clc
2504                            <1> gpa_retn:
2505 0000637B C3                <1>       retn
2506                            <1> gpa_7:
2507 0000637C E85AF8FFFF          <1>       call   allocate_page
2508 00006381 72CE              <1>       jc     short gpa_im_err ; 'insufficient memory' error
2509 00006383 E8CDF8FFFF          <1>       call   clear_page
2510 00006388 EBA7              <1>       jmp    short gpa_2
2511                            <1>
2512                            <1> gpa_8: ; 26/03/2017
2513 0000638A C605[EA9B0100]00    <1>       mov    byte [no_page_swap], 0
2514 00006391 53                <1>       push   ebx
2515 00006392 50                <1>       push   eax ; 26/03/2017
2516 00006393 6681E300F0          <1>         and     bx, ~PAGE_OFF ; ~0FFFh ; reset bits, 0 to 11
2517 00006398 8A1D[B3030300]      <1>       mov    bl, [u.uno] ; current process number
2518 0000639E E89DFDFFFF          <1>       call   swap_queue_shift ; drop from the queue if
2519                            <1>                               ; it is already on the queue
2520 000063A3 58                <1>       pop    eax ; 26/03/2017
2521 000063A4 5B                <1>       pop    ebx
2522 000063A5 C3                <1>       retn
2523                            <1>
2524                            <1> reload_page:
2525                            <1>       ; 20/07/2015
2526                            <1>       ; 28/04/2015 (Retro UNIX 386 v1 - beginning)
2527                            <1>       ;
2528                            <1>       ; Reload (Restore) swapped page at memory
2529                            <1>       ;
2530                            <1>       ; INPUT ->
2531                            <1>       ;      EBP = Virtual (linear) memory address
2532                            <1>       ;      EAX = PTE value (swap disk sector address)
2533                            <1>       ;      (Swap disk sector address = bit 1 to bit 31 of EAX)
2534                            <1>       ; OUTPUT ->
2535                            <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS OF RELOADED PAGE
2536                            <1>       ;
2537                            <1>       ;      CF = 1 and EAX = error code
2538                            <1>       ;
2539                            <1>       ; Modified Registers -> none (except EAX)
2540                            <1>       ;
2541 000063A6 D1E8              <1>       shr    eax, 1   ; Convert PTE value to swap disk address
2542 000063A8 53                <1>       push   ebx      ;
2543 000063A9 89C3              <1>       mov    ebx, eax ; Swap disk (offset) address
2544 000063AB E82BF8FFFF          <1>       call   allocate_page
2545 000063B0 720C              <1>       jc     short rlp_im_err
2546 000063B2 93                <1>       xchg   eax, ebx
2547                            <1>       ; EBX = Physical memory (page) address
2548                            <1>       ; EAX = Swap disk (offset) address
2549                            <1>       ; EBP = Virtual (linear) memory address
```

```
2550 000063B3 E862FCFFFF          <1>       call   swap_in
2551 000063B8 720B                <1>       jc     short rlp_swp_err  ; (swap disk/file read error)
2552 000063BA 89D8                <1>       mov    eax, ebx
2553                              <1> rlp_retn:
2554 000063BC 5B                  <1>       pop    ebx
2555 000063BD C3                  <1>       retn
2556                              <1>
2557                              <1> rlp_im_err:
2558 000063BE B804000000          <1>       mov    eax, ERR_MINOR_IM ; Insufficient memory (minor) error!
2559                              <1>                                ; Major error = 0 (No protection fault)
2560 000063C3 EBF7                <1>       jmp    short rlp_retn
2561                              <1>
2562                              <1> rlp_swp_err:
2563 000063C5 B828000000          <1>       mov    eax, SWP_DISK_READ_ERR ; Swap disk read error !
2564 000063CA EBF0                <1>       jmp    short rlp_retn
2565                              <1>
2566                              <1>
2567                              <1> copy_page_dir:
2568                              <1>       ; 19/09/2015
2569                              <1>       ; temporary - 07/09/2015
2570                              <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
2571                              <1>       ;
2572                              <1>       ; INPUT ->
2573                              <1>       ;      [u.pgdir] = PHYSICAL (real/flat) ADDRESS of the parent's
2574                              <1>       ;                  page directory.
2575                              <1>       ; OUTPUT ->
2576                              <1>       ;      EAX =  PHYSICAL (real/flat) ADDRESS of the child's
2577                              <1>       ;             page directory.
2578                              <1>       ;      (New page directory with new page table entries.)
2579                              <1>       ;      (New page tables with read only copies of the parent's
2580                              <1>       ;      pages.)
2581                              <1>       ;      EAX = 0 -> Error (CF = 1)
2582                              <1>       ;
2583                              <1>       ; Modified Registers -> none (except EAX)
2584                              <1>       ;
2585 000063CC E80AF8FFFF          <1>       call   allocate_page
2586 000063D1 723E                <1>       jc     short cpd_err
2587                              <1>       ;
2588 000063D3 55                  <1>       push   ebp ; 20/07/2015
2589 000063D4 56                  <1>       push   esi
2590 000063D5 57                  <1>       push   edi
2591 000063D6 53                  <1>       push   ebx
2592 000063D7 51                  <1>       push   ecx
2593 000063D8 8B35[B8030300]      <1>       mov    esi, [u.pgdir]
2594 000063DE 89C7                <1>       mov    edi, eax
2595 000063E0 50                  <1>       push   eax ; save child's page directory address
2596                              <1>       ; copy PDE 0 from the parent's page dir to the child's page dir
2597                              <1>       ; (use same system space for all user page tables)
2598 000063E1 A5                  <1>       movsd
2599 000063E2 BD00004000          <1>       mov    ebp, 1024*4096 ; pass the 1st 4MB (system space)
2600 000063E7 B9FF030000          <1>       mov    ecx, (PAGE_SIZE / 4) - 1 ; 1023
2601                              <1> cpd_0:
2602 000063EC AD                  <1>       lodsd
2603                              <1>       ;or    eax, eax
2604                              <1>         ;jnz     short cpd_1
2605 000063ED A801                <1>       test   al, PDE_A_PRESENT ;  bit 0 =  1
2606 000063EF 7508                <1>       jnz    short cpd_1
2607                              <1>       ; (virtual address at the end of the page table)
2608 000063F1 81C500004000        <1>       add    ebp, 1024*4096 ; page size * PTE count
2609 000063F7 EB0F                <1>       jmp    short cpd_2
2610                              <1> cpd_1:
2611 000063F9 662500F0            <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear attribute bits
2612 000063FD 89C3                <1>       mov    ebx, eax
2613                              <1>       ; EBX = Parent's page table address
2614 000063FF E81F000000          <1>       call   copy_page_table
2615 00006404 720C                <1>       jc     short cpd_p_err
2616                              <1>       ; EAX = Child's page table address
2617 00006406 0C07                <1>       or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
2618                              <1>                      ; set bit 0, bit 1 and bit 2 to 1
2619                              <1>                      ; (present, writable, user)
2620                              <1> cpd_2:
2621 00006408 AB                  <1>       stosd
2622 00006409 E2E1                <1>       loop   cpd_0
2623                              <1>       ;
2624 0000640B 58                  <1>       pop    eax  ; restore child's page directory address
2625                              <1> cpd_3:
2626 0000640C 59                  <1>       pop    ecx
2627 0000640D 5B                  <1>       pop    ebx
2628 0000640E 5F                  <1>       pop    edi
2629 0000640F 5E                  <1>       pop    esi
2630 00006410 5D                  <1>       pop    ebp
2631                              <1> cpd_err:
2632 00006411 C3                  <1>       retn
2633                              <1> cpd_p_err:
2634                              <1>       ; release the allocated pages missing (recover free space)
2635 00006412 58                  <1>       pop    eax  ; the new page directory address (physical)
2636 00006413 8B1D[B8030300]      <1>       mov    ebx, [u.pgdir] ; parent's page directory address
2637 00006419 E8F6F8FFFF          <1>       call   deallocate_page_dir
2638 0000641E 29C0                <1>       sub    eax, eax ; 0
2639 00006420 F9                  <1>       stc
2640 00006421 EBE9                <1>       jmp    short cpd_3
2641                              <1>
2642                              <1> copy_page_table:
2643                              <1>       ; 19/09/2015
2644                              <1>       ; temporary - 07/09/2015
2645                              <1>       ; 07/09/2015 (Retro UNIX 386 v1 - beginning)
2646                              <1>       ;
2647                              <1>       ; INPUT ->
2648                              <1>       ;      EBX = PHYSICAL (real/flat) ADDRESS of the parent's page table.
2649                              <1>       ;      EBP = page table entry index (from 'copy_page_dir')
2650                              <1>       ; OUTPUT ->
2651                              <1>       ;      EAX = PHYSICAL (real/flat) ADDRESS of the child's page table.
2652                              <1>       ;      EBP = (recent) page table index (for 'add_to_swap_queue')
2653                              <1>       ;      CF = 1 -> error
2654                              <1>       ;
```

```
2655                            <1>       ; Modified Registers -> EBP (except EAX)
2656                            <1>       ;
2657 00006423 E8B3F7FFFF      <1>       call   allocate_page
2658 00006428 725A             <1>       jc     short cpt_err
2659                            <1>       ;
2660 0000642A 50               <1>       push   eax ; *
2661                            <1>       ;push  ebx
2662 0000642B 56               <1>       push   esi
2663 0000642C 57               <1>       push   edi
2664 0000642D 52               <1>       push   edx
2665 0000642E 51               <1>       push   ecx
2666                            <1>       ;
2667 0000642F 89DE             <1>       mov    esi, ebx
2668 00006431 89C7             <1>       mov    edi, eax
2669 00006433 89C2             <1>       mov    edx, eax
2670 00006435 81C200100000     <1>       add    edx, PAGE_SIZE
2671                            <1> cpt_0:
2672 0000643B AD               <1>       lodsd
2673 0000643C A801             <1>       test   al, PTE_A_PRESENT ;  bit 0 =  1
2674 0000643E 750B             <1>       jnz    short cpt_1
2675 00006440 21C0             <1>       and    eax, eax
2676 00006442 7430             <1>       jz     short cpt_2
2677                            <1>       ; ebp = virtual (linear) address of the memory page
2678 00006444 E85DFFFFFF       <1>       call   reload_page ; 28/04/2015
2679 00006449 7234             <1>       jc     short cpt_p_err
2680                            <1> cpt_1:
2681 0000644B 662500F0         <1>       and    ax, PTE_A_CLEAR ; 0F000h ; clear attribute bits
2682 0000644F 89C1             <1>       mov    ecx, eax
2683                            <1>       ; Allocate a new page for the child process
2684 00006451 E885F7FFFF       <1>       call   allocate_page
2685 00006456 7227             <1>       jc     short cpt_p_err
2686 00006458 57               <1>       push   edi
2687 00006459 56               <1>       push   esi
2688 0000645A 89CE             <1>       mov    esi, ecx
2689 0000645C 89C7             <1>       mov    edi, eax
2690 0000645E B900040000       <1>       mov    ecx, PAGE_SIZE/4
2691 00006463 F3A5             <1>       rep    movsd  ; copy page (4096 bytes)
2692 00006465 5E               <1>       pop    esi
2693 00006466 5F               <1>       pop    edi
2694                            <1>       ;
2695 00006467 53               <1>       push   ebx
2696 00006468 50               <1>       push   eax
2697 00006469 89EB             <1>       mov    ebx, ebp
2698                            <1>       ; ebx = virtual address of the memory page
2699 0000646B E8AAFDFFFF       <1>       call   add_to_swap_queue
2700 00006470 58               <1>       pop    eax
2701 00006471 5B               <1>       pop    ebx
2702                            <1>       ;
2703                            <1>       ;or    ax, PTE_A_USER+PTE_A_PRESENT
2704 00006472 0C07             <1>       or     al, PTE_A_USER+PTE_A_WRITE+PTE_A_PRESENT
2705                            <1> cpt_2:
2706 00006474 AB               <1>       stosd  ; EDI points to child's PTE
2707                            <1>       ;
2708 00006475 81C500100000     <1>       add    ebp, 4096 ; 20/07/2015 (next page)
2709                            <1>       ;
2710 0000647B 39D7             <1>       cmp    edi, edx
2711 0000647D 72BC             <1>       jb     short cpt_0
2712                            <1> cpt_p_err:
2713 0000647F 59               <1>       pop    ecx
2714 00006480 5A               <1>       pop    edx
2715 00006481 5F               <1>       pop    edi
2716 00006482 5E               <1>       pop    esi
2717                            <1>       ;pop   ebx
2718 00006483 58               <1>       pop    eax ; *
2719                            <1> cpt_err:
2720 00006484 C3               <1>       retn
2721                            <1>
2722                            <1> allocate_memory_block:
2723                            <1>       ; 01/05/2017
2724                            <1>       ; 28/04/2017
2725                            <1>       ; 25/04/2017
2726                            <1>       ; 01/04/2016, 02/04/2016, 03/04/2016
2727                            <1>       ; 13/03/2016, 14/03/2016
2728                            <1>       ; 12/03/2016 (TRDOS 386 = TRDOS v2.0)
2729                            <1>       ; Allocating contiguous memory pages (in the kernel's memory space)
2730                            <1>       ;
2731                            <1>       ; INPUT ->
2732                            <1>       ;     EAX = Beginning address (physical)
2733                            <1>       ;     EAX = 0 -> Allocate memory block from the first proper aperture
2734                            <1>       ;     ECX = Number of bytes to be allocated
2735                            <1>       ;
2736                            <1>       ; OUTPUT ->
2737                            <1>       ;     1) cf = 0 -> successful
2738                            <1>       ;     EAX = Beginning (physical) address of the allocated memory block
2739                            <1>       ;     ECX = Number of allocated bytes (rounded up to page borders)
2740                            <1>       ;     2) cf = 1 -> unsuccessful
2741                            <1>       ;      2.1) If EAX > 0 ->
2742                            <1>       ;           (Number of requested pages is more than # of free pages
2743                            <1>       ;            but contiguous free pages -the aperture- is not enough!)
2744                            <1>       ;           EAX = Beginning address of available aperture
2745                            <1>       ;                (one of all aperture with max. aperture size/length)
2746                            <1>       ;           ECX = Size of available aperture (memory block) in bytes
2747                            <1>       ;      2.2) If EAX = 0 -> Out of memory error
2748                            <1>       ;                 (number of free pages is less than requested number)
2749                            <1>       ;           ECX = Total number of free bytes (free pages * 4096)
2750                            <1>       ;                (It is not number of contiguous free bytes)
2751                            <1>       ;
2752                            <1>       ; (Modified Registers -> EAX, ECX)
2753                            <1>       ;
2754                            <1>       ; PURPOSE: Loading a file at memory for copying or running etc.
2755                            <1>       ; If this procedure returns with cf is set, ECX contains maximum
2756                            <1>       ; available space and EAX contains the beginning address of it.
2757                            <1>       ; If EAX has zero, ECX contains total number of free bytes.
2758                            <1>       ; If requested block has been successfully allocated (by rounding up to
2759                            <1>       ; the last page border), it must be deallocated later by using
```

```
                                <1>       ; 'deallocate_memory_block' procedure.
                                <1>
00006485 52                     <1>       push   edx ; *
00006486 BAFF0F0000             <1>       mov    edx, PAGE_SIZE - 1   ; 4095
0000648B 01D0                   <1>       add    eax, edx
0000648D 01D1                   <1>       add    ecx, edx
0000648F C1E90C                 <1>       shr    ecx, PAGE_SHIFT          ; 12
                                <1>
                                <1>       ; ECX = number of contiguous pages to be allocated
00006492 8B15[208A0100]         <1>       mov    edx, [free_pages]
                                <1>       ; 01/05/2017
                                <1>       ;or    ecx, ecx
                                <1>       ;jz    short amb3
                                <1>       ; If ECX=0, set cf to 1 and return with max. available mem block size
                                <1>
00006498 39D1                   <1>       cmp    ecx, edx
0000649A 7760                   <1>       ja     short amb_3
                                <1>
0000649C C1E80C                 <1>       shr    eax, PAGE_SHIFT      ; 12
                                <1>
0000649F 89C2                   <1>       mov    edx, eax          ; page number
000064A1 C1EA03                 <1>       shr    edx, 3            ; to get offset to M.A.T.
                                <1>                                ; (1 allocation bit = 1 page)
                                <1>                                ; (1 allocation bytes = 8 pages)
000064A4 80E2FC                 <1>       and    dl, 0FCh          ; clear lower 2 bits
                                <1>                                ; (to get 32 bit position)
000064A7 53                     <1>       push   ebx ; **
                                <1> amb_0:
000064A8 890D[D4950100]         <1>       mov    [mem_ipg_count], ecx ; initial (reset) value of page count
000064AE 890D[D8950100]         <1>       mov    [mem_pg_count], ecx
000064B4 31C9                   <1>       xor    ecx, ecx ; 0
000064B6 890D[DC950100]         <1>       mov    [mem_aperture], ecx ; 0
000064BC 890D[E0950100]         <1>       mov    [mem_max_aperture], ecx ; 0
                                <1>
000064C2 BB00001000             <1>       mov    ebx, MEM_ALLOC_TBL   ; Memory Allocation Table address.
000064C7 3B15[248A0100]         <1>       cmp    edx, [next_page]     ; Is the beginning page address lower
                                <1>                                ; than the address in 'next_page' ?
                                <1>                                ; (the first/next free page of user space)
000064CD 7208                   <1>       jb     short amb_1
000064CF 3B15[288A0100]         <1>       cmp    edx, [last_page]     ; is the beginning page address higher
                                <1>                                ; than the address in 'last_page' ?
                                <1>                                ; (end of the memory)
000064D5 7606                   <1>       jna    short amb_2       ; no
                                <1> amb_1:
000064D7 8B15[248A0100]         <1>       mov    edx, [next_page]     ; M.A.T. offset (1 M.A.T. byte = 8 pages)
                                <1> amb_2:
000064DD 01D3                   <1>       add    ebx, edx
                                <1>
                                <1>       ; 28/04/2017
                                <1>       ;xor   ecx, ecx
000064DF 0FBC0B                 <1>       bsf    ecx, [ebx]        ; 0 to 31
000064E2 89D0                   <1>       mov    eax, edx
000064E4 C1E003                 <1>       shl    eax, 3            ; *8
000064E7 01C8                   <1>       add    eax, ecx          ; beginning page number
                                <1>
000064E9 A3[E4950100]           <1>       mov    [mem_pg_pos], eax    ; beginning page no (for curr. mem. aperture)
000064EE A3[E8950100]           <1>       mov    [mem_max_pg_pos], eax ; beginning page no for max. mem. aperture
                                <1>
000064F3 83E01F                 <1>       and    eax, 1Fh          ; lower 5 bits only (0 to 31)
                                <1>                                ; (allocation bit position)
000064F6 750E                   <1>       jnz    short amb_4       ; 0
000064F8 B120                   <1>       mov    cl, 32
000064FA EB4B                   <1>       jmp    short amb_10
                                <1>
                                <1> amb_3:       ; out_of_memory
000064FC 31C0                   <1>       xor    eax, eax ; 0
000064FE 89D1                   <1>       mov    ecx, edx ; free pages
00006500 C1E10C                 <1>       shl    ecx, PAGE_SHIFT
00006503 5A                     <1>       pop    edx ; *
00006504 F9                     <1>       stc
00006505 C3                     <1>       retn
                                <1> amb_4:
00006506 8B13                   <1>       mov    edx, [ebx]
00006508 88C1                   <1>       mov    cl, al ; 1 to 31
0000650A D3EA                   <1>       shr    edx, cl
0000650C 89D0                   <1>       mov    eax, edx
                                <1> amb_5:
0000650E D1E8                   <1>       shr    eax, 1 ; (***)
00006510 7317                   <1>       jnc    short amb_7
00006512 FF05[DC950100]         <1>       inc    dword [mem_aperture]
00006518 FF0D[D8950100]         <1>       dec    dword [mem_pg_count]
0000651E 7470                   <1>       jz     short amb_15
                                <1> amb_6:
                                <1>       ; 28/04/2017
00006520 FEC1                   <1>       inc    cl
00006522 80F920                 <1>       cmp    cl, 32
00006525 730D                   <1>       jnb    short amb_9
00006527 EBE5                   <1>       jmp    short amb_5
                                <1> amb_7:
00006529 50                     <1>       push   eax ; (***) allocation bits (in shifted status)
0000652A E81B010000             <1>       call   amb_26 ; set maximum memory aperture (free memory block size)
0000652F 58                     <1>       pop    eax ; (***)
00006530 EBEE                   <1>       jmp    short amb_6
                                <1> amb_8:
                                <1>       ; 28/04/2017
00006532 B120                   <1>       mov    cl, 32
                                <1> amb_9:
00006534 89DA                   <1>       mov    edx, ebx
00006536 81EA00001000           <1>       sub    edx, MEM_ALLOC_TBL
0000653C 3B15[288A0100]         <1>       cmp    edx, [last_page]
00006542 7336                   <1>       jnb    short amb_14 ; contiguous pages not enough
00006544 83C304                 <1>       add    ebx, 4
                                <1> amb_10:
00006547 8B03                   <1>       mov    eax, [ebx]
00006549 21C0                   <1>       and    eax, eax
```

```
2865 0000654B 7408              <1>         jz      short amb_11 ; there is not a free page bit in this alloc dword
2866 0000654D 40                <1>       inc     eax ; 0FFFFFFFFh -> 0
2867 0000654E 740C              <1>       jz      short amb_12 ; all of bits are set (32 free pages)
2868 00006550 48                <1>       dec     eax
2869 00006551 28C9              <1>       sub     cl, cl ; 0
2870 00006553 EBB9              <1>       jmp     short amb_5
2871                            <1> amb_11:
2872 00006555 E8F0000000        <1>       call    amb_26 ; set maximum memory aperture (free memory block size)
2873 0000655A EBD8              <1>       jmp     short amb_9
2874                            <1> amb_12:
2875 0000655C 390D[D8950100]    <1>       cmp     [mem_pg_count], ecx ; 32
2876 00006562 7306              <1>       jnb     short amb_13
2877 00006564 8B0D[D8950100]    <1>       mov     ecx, [mem_pg_count]
2878                            <1> amb_13:
2879 0000656A 010D[DC950100]    <1>       add     [mem_aperture], ecx
2880 00006570 290D[D8950100]    <1>       sub     [mem_pg_count], ecx
2881 00006576 7618              <1>       jna     short amb_15
2882 00006578 EBBA              <1>       jmp     short amb_9 ; 01/05/2017
2883                            <1> amb_14:
2884 0000657A E8CB000000        <1>       call    amb_26 ; 28/04/2017
2885 0000657F A1[E8950100]      <1>       mov     eax, [mem_max_pg_pos] ; begin address of max. mem aperture
2886 00006584 8B0D[E0950100]    <1>       mov     ecx, [mem_max_aperture] ; max. (largest) memory aperture
2887 0000658A F9                <1>       stc
2888 0000658B E9AF000000        <1>         jmp     amb_25
2889                            <1>
2890                            <1> amb_15: ; OK !
2891 00006590 A1[E4950100]      <1>       mov     eax, [mem_pg_pos]     ; Beginning address as page number
2892 00006595 8B0D[DC950100]    <1>       mov     ecx, [mem_aperture]   ; Free contiguous page count (>=1)
2893                            <1> amb_16:
2894                            <1>       ; allocate contiguous memory pages (via memory allocation table bits)
2895 0000659B 89C2              <1>       mov     edx, eax
2896                            <1>       ; 25/04/2017
2897 0000659D C1EA03            <1>       shr     edx, 3        ; 8 pages in one allocation byte
2898 000065A0 80E2FC            <1>       and     dl, 0FCh      ; clear lower 2 bits
2899                            <1>                             ; (for dword/32bit positioning)
2900                            <1>
2901 000065A3 BB00001000        <1>       mov     ebx, MEM_ALLOC_TBL
2902 000065A8 01D3              <1>       add     ebx, edx
2903 000065AA 83E01F            <1>       and     eax, 1Fh ; 31
2904                            <1>       ; 03/04/2016
2905 000065AD BA20000000        <1>       mov     edx, 32
2906 000065B2 28C2              <1>       sub     dl, al
2907 000065B4 39CA              <1>       cmp     edx, ecx      ; ecx >= 1
2908 000065B6 7602              <1>       jna     short amb_17
2909 000065B8 89CA              <1>       mov     edx, ecx
2910                            <1> amb_17:
2911 000065BA 29D1              <1>       sub     ecx, edx
2912 000065BC 51                <1>       push    ecx ; ***
2913 000065BD 89D1              <1>       mov     ecx, edx
2914                            <1> amb_18:
2915 000065BF 0FB303            <1>       btr     [ebx], eax    ; The destination bit indexed by the source value
2916                            <1>                             ; is copied into the Carry Flag and then cleared
2917                            <1>                             ; in the destination.
2918 000065C2 FF0D[208A0100]    <1>       dec     dword [free_pages] ; 1 page has been allocated (X = X-1)
2919 000065C8 49                <1>       dec     ecx
2920 000065C9 7404              <1>       jz      short amb_19
2921 000065CB FEC0              <1>       inc     al
2922 000065CD EBF0              <1>       jmp     short amb_18
2923                            <1> amb_19:
2924 000065CF 59                <1>       pop     ecx ; ***
2925 000065D0 21C9              <1>       and     ecx, ecx ; 0 ?
2926 000065D2 741E              <1>       jz      short amb_22
2927                            <1>       ; 01/04/2016
2928 000065D4 B020              <1>       mov     al, 32
2929                            <1> amb_20:
2930 000065D6 83C304            <1>       add     ebx, 4
2931 000065D9 39C1              <1>       cmp     ecx, eax ; 32
2932 000065DB 7305              <1>       jnb     short amb_21
2933                            <1>       ; ECX < 32
2934 000065DD 28C0              <1>       sub     al, al ; 0
2935 000065DF 50                <1>       push    eax ; 0 ***
2936 000065E0 EBDD              <1>       jmp     short amb_18
2937                            <1> amb_21:
2938 000065E2 2905[208A0100]    <1>       sub     [free_pages], eax   ; [free_pages] = [free_pages] - 32
2939 000065E8 C70300000000      <1>       mov     dword [ebx], 0        ; reset 32 bits
2940 000065EE 29C1              <1>       sub     ecx, eax ; 32
2941 000065F0 75E4              <1>       jnz     short amb_20
2942                            <1> amb_22:
2943 000065F2 A1[E4950100]      <1>       mov     eax, [mem_pg_pos]   ; Beginning address as page number
2944 000065F7 8B0D[DC950100]    <1>       mov     ecx, [mem_aperture] ; Free contiguous page count
2945                            <1>       ; [next_page] update
2946 000065FD 89C2              <1>       mov     edx, eax
2947                            <1>       ; 03/04/2016
2948 000065FF C1EA03            <1>       shr     edx, 3            ; to get offset to M.A.T.
2949                            <1>                                 ; (1 allocation bit = 1 page)
2950                            <1>                                 ; (1 allocation bytes = 8 pages)
2951 00006602 80E2FC            <1>       and     dl, 0FCh          ; clear lower 2 bits
2952                            <1>                                 ; (to get 32 bit position)
2953 00006605 3B15[248A0100]    <1>       cmp     edx, [next_page] ; first free page pointer offset
2954 0000660B 7732              <1>       ja      short amb_25
2955 0000660D BB00001000        <1>       mov     ebx, MEM_ALLOC_TBL
2956 00006612 833C1300          <1>       cmp     dword [ebx+edx], 0
2957 00006616 7721              <1>       ja      short amb_24
2958 00006618 89C2              <1>       mov     edx, eax
2959 0000661A 01CA              <1>       add     edx, ecx
2960 0000661C C1EA03            <1>       shr     edx, 3
2961 0000661F 80E2FC            <1>       and     dl, 0FCh
2962                            <1> amb_23:
2963 00006622 833C1300          <1>       cmp     dword [ebx+edx], 0
2964 00006626 7711              <1>       ja      short amb_24
2965 00006628 83C204            <1>       add     edx, 4
2966 0000662B 3B15[288A0100]    <1>       cmp     edx, [last_page]    ; last page pointer offset
2967 00006631 76EF              <1>       jna     short amb_23
2968 00006633 8B15[2C8A0100]    <1>       mov     edx, [first_page]   ; (for) beginning of user's space
2969                            <1> amb_24:
```

```
2970 00006639 8915[248A0100]      <1>       mov    [next_page], edx
2971                              <1> amb_25:
2972 0000663F 9C                  <1>       pushf
2973 00006640 C1E00C              <1>       shl    eax, PAGE_SHIFT          ; convert to phy. address in bytes
2974 00006643 C1E10C              <1>       shl    ecx, PAGE_SHIFT          ; convert to byte counts
2975 00006646 9D                  <1>       popf
2976 00006647 5B                  <1>       pop    ebx ; **
2977 00006648 5A                  <1>       pop    edx ; *
2978 00006649 C3                  <1>       retn
2979                              <1>
2980                              <1> amb_26:     ; set maximum free memory aperture (free memory block size)
2981 0000664A 89DA                <1>       mov    edx, ebx ; current address
2982 0000664C 81EA00001000        <1>       sub    edx, MEM_ALLOC_TBL ; MAT beginning address
2983                              <1>       ; 02/04/2016
2984 00006652 C1E203              <1>       shl    edx, 3 ; MAT byte offset * 8 = page number base
2985 00006655 01CA                <1>       add    edx, ecx ; current page number (ecx = 0 to 32)
2986                              <1>       ;
2987 00006657 A1[DC950100]        <1>       mov    eax, [mem_aperture]
2988 0000665C 21C0                <1>       and    eax, eax
2989 0000665E 7421                <1>         jz     short amb_27
2990 00006660 C705[DC950100]0000- <1>         mov    dword [mem_aperture], 0
2990 00006668 0000                <1>
2991 0000666A 3B05[E0950100]      <1>       cmp    eax, [mem_max_aperture]
2992 00006670 760F                <1>       jna    short amb_27
2993 00006672 A3[E0950100]        <1>       mov    [mem_max_aperture], eax
2994                              <1>       ; 25/04/2017
2995 00006677 A1[E4950100]        <1>       mov    eax, [mem_pg_pos]
2996                              <1>       ; EAX = Beginning page number of the max. aperture
2997 0000667C A3[E8950100]        <1>       mov    [mem_max_pg_pos], eax
2998                              <1> amb_27:
2999 00006681 8915[E4950100]      <1>       mov    [mem_pg_pos], edx ; current page
3000                              <1>
3001 00006687 A1[D4950100]        <1>       mov    eax, [mem_ipg_count] ; initial (reset) value of page count
3002 0000668C A3[D8950100]        <1>       mov    [mem_pg_count], eax
3003                              <1>
3004 00006691 C3                  <1>       retn
3005                              <1>
3006                              <1> deallocate_memory_block:
3007                              <1>       ; 03/04/2016
3008                              <1>       ; 14/03/2016 (TRDOS 386 = TRDOS v2.0)
3009                              <1>       ; Deallocating contiguous memory pages (in the kernel's memory space)
3010                              <1>       ;
3011                              <1>       ; INPUT ->
3012                              <1>       ;     EAX = Beginning address (physical)
3013                              <1>       ;     ECX = Number of bytes to be deallocated
3014                              <1>       ;
3015                              <1>       ; OUTPUT ->
3016                              <1>       ;     Memory Allocation Table bits will be updated
3017                              <1>       ;     [free_pages] will be changed (increased)
3018                              <1>       ;
3019                              <1>       ; (Modified Registers -> EAX, ECX)
3020                              <1>       ;
3021                              <1>       ; PURPOSE: Unloading/Freeing a file -or an allocated memory block-
3022                              <1>       ; at memory after copying, running, saving, reading, writing etc.
3023                              <1>       ;
3024                              <1>
3025 00006692 52                  <1>       push   edx ; *
3026 00006693 53                  <1>       push   ebx ; **
3027                              <1>
3028 00006694 C1E80C              <1>       shr    eax, PAGE_SHIFT          ; 12
3029 00006697 C1E90C              <1>       shr    ecx, PAGE_SHIFT          ; 12
3030                              <1>
3031                              <1>       ; EAX = Beginning page number
3032                              <1>       ; ECX = Number of contiguous pages to be deallocated
3033                              <1> damb_0:
3034                              <1>       ; deallocate contiguous memory pages (via memory allocation table bits)
3035 0000669A 89C2                <1>       mov    edx, eax
3036 0000669C C1EA03              <1>       shr    edx, 3            ; to get offset to M.A.T.
3037                              <1>                                ; (1 allocation bit = 1 page)
3038                              <1>                                ; (1 allocation bytes = 8 pages)
3039 0000669F 80E2FC              <1>       and    dl, 0FCh          ; clear lower 2 bits
3040                              <1>                                ; (to get 32 bit position)
3041 000066A2 3B15[248A0100]      <1>       cmp    edx, [next_page] ; next free page
3042 000066A8 7306                <1>       jnb    short damb_1
3043 000066AA 8915[248A0100]      <1>       mov    [next_page], edx
3044                              <1> damb_1:
3045 000066B0 BB00001000          <1>       mov    ebx, MEM_ALLOC_TBL
3046 000066B5 01D3                <1>       add    ebx, edx
3047 000066B7 83E01F              <1>       and    eax, 1Fh ; 31
3048                              <1>
3049                              <1>       ; 03/04/2016
3050 000066BA BA20000000          <1>       mov    edx, 32
3051 000066BF 28C2                <1>       sub    dl, al
3052 000066C1 39CA                <1>       cmp    edx, ecx
3053 000066C3 7602                <1>       jna    short damb_2
3054 000066C5 89CA                <1>       mov    edx, ecx
3055                              <1> damb_2:
3056 000066C7 29D1                <1>       sub    ecx, edx
3057 000066C9 51                  <1>       push   ecx ; ***
3058 000066CA 89D1                <1>       mov    ecx, edx
3059                              <1> damb_3:
3060 000066CC 0FAB03              <1>       bts    [ebx], eax        ; unlink/release/deallocate page
3061                              <1>                                ; set relevant bit to 1.
3062                              <1>                                ; set CF to the previous bit value
3063 000066CF FF05[208A0100]      <1>       inc     dword [free_pages]   ; 1 page has been deallocated (X = X+1)
3064 000066D5 49                  <1>       dec    ecx
3065 000066D6 7404                <1>       jz     short damb_4
3066 000066D8 FEC0                <1>       inc    al
3067 000066DA EBF0                <1>       jmp    short damb_3
3068                              <1> damb_4:
3069 000066DC 59                  <1>       pop    ecx ; ***
3070 000066DD 21C9                <1>       and    ecx, ecx ; 0 ?
3071 000066DF 741E                <1>       jz     short damb_7
3072                              <1>       ; 03/04/2016
3073 000066E1 B020                <1>       mov    al, 32
```

```
                                <1> damb_5:
000066E3 83C304                 <1>         add     ebx, 4
000066E6 39C1                   <1>         cmp     ecx, eax ; 32
000066E8 7305                   <1>         jnb     short damb_6
                                <1>         ; ECX < 32
000066EA 28C0                   <1>         sub     al, al ; 0
000066EC 50                     <1>         push    eax ; 0 ***
000066ED EBDD                   <1>         jmp     short damb_3
                                <1> damb_6:
000066EF 0105[208A0100]         <1>         add     [free_pages], eax ; [free_pages] = [free_pages] + 32
000066F5 C703FFFFFFFF           <1>         mov     dword [ebx], 0FFFFFFFFh ; set 32 bits
000066FB 29C1                   <1>         sub     ecx, eax ; 32
000066FD 75E4                   <1>         jnz     short damb_5
                                <1> damb_7:
000066FF 5B                     <1>         pop     ebx ; **
00006700 5A                     <1>         pop     edx ; *
00006701 C3                     <1>         retn
                                <1>
                                <1> direct_memory_access:
                                <1>         ; 22/07/2017
                                <1>         ; 12/05/2017
                                <1>         ; 16/07/2016
                                <1>         ; 12/07/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; This processure will be called to map
                                <1>         ; user's (ring 3) page tables to access phsical
                                <1>         ; (flat/linear) memory addresses, directly (without
                                <1>         ;  kernel's data transfer functions).
                                <1>         ;
                                <1>         ; Purpose: Video memory access and shared memory access.
                                <1>         ;
                                <1>         ; INPUT ->
                                <1>         ;       EAX = Beginning address (physical).
                                <1>         ;       EBX = User's buffer address ; 12/05/2017
                                <1>         ;       ECX = Number of contiguous pages to be mapped.
                                <1>         ; OUTPUT ->
                                <1>         ;       User's page directory and pages tables
                                <1>         ;       will be updated.
                                <1>         ;
                                <1>         ;       If an old page table entry has valid page address,
                                <1>         ;       that page will be deallocated just before PTE will
                                <1>         ;       be changed for direct (1 to 1) memory page access.
                                <1>         ;
                                <1>         ;       If old PTE value points to a swapped page,
                                <1>           ;         that page (block) will be unlinked on swap disk.
                                <1>         ;
                                <1>         ;       Newly allocated pages (except page tables) will not
                                <1>         ;       be applied to Memory Allocation Table.
                                <1>         ;       AVL bit 1 (PTE bit 10) of page table entry will be
                                <1>         ;       used to indicate shared (direct) memory page; then,
                                <1>         ;       this page will not be deallocated later during
                                <1>         ;       process termination. (Memory Allocation Table and
                                <1>         ;       free memory count will not be affected.
                                <1>         ;       (Except deallocating page table's itself.)
                                <1>         ;
                                <1>         ;        CF = 1 -> error (EAX = error code)
                                <1>         ;        CF = 0 -> success (EAX = beginning address)
                                <1>         ;
                                <1>         ;; (Modified Registers -> none)
                                <1>         ; Modified registers: ebp, edx, ecx, ebx, esi, edi
                                <1>         ;
                                <1>
                                <1>         ;push  ebp
                                <1>         ;push  ebx
                                <1>         ;push  ecx
                                <1>         ;push  edx
00006702 662500F0               <1>         and     ax, PTE_A_CLEAR ; clear page offset
00006706 50                     <1>         push    eax
                                <1>         ;and   ecx, ecx ; page count
                                <1>         ;jz    dmem_acc_7  ; 'insufficient memory' error
00006707 89C5                   <1>         mov     ebp, eax
00006709 81C300004000           <1>         add     ebx, CORE ; 12/05/2017
                                <1> dmem_acc_0:
0000670F 891D[D4A00100]         <1>         mov     [base_addr], ebx ; 12/05/2017
00006715 A1[B8030300]           <1>         mov     eax, [u.pgdir] ; page dir address (physical)
0000671A E8D7F5FFFF             <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 ?
0000671F 7324                   <1>         jnc     short dmem_acc_1
                                <1>         ;
00006721 E8B5F4FFFF             <1>         call    allocate_page
00006726 0F82AB000000           <1>           jc      dmem_acc_7  ; 'insufficient memory' error
                                <1>         ;
0000672C E824F5FFFF             <1>         call    clear_page
                                <1>         ; EAX = Physical (base) address of the allocated (new) page
00006731 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)
00006733 8902                   <1>         mov     [edx], eax ; Let's put the new page directory entry here !
00006735 A1[B8030300]           <1>         mov     eax, [u.pgdir]
0000673A E8B7F5FFFF             <1>         call    get_pte
0000673F 0F8292000000           <1>           jc      dmem_acc_7 ; 'insufficient memory' error
                                <1> dmem_acc_1:
                                <1>         ; EAX = PTE value, EDX = PTE address
00006745 A801                   <1>         test    al, PTE_A_PRESENT
00006747 750D                   <1>         jnz     short dmem_acc_2
00006749 09C0                   <1>         or      eax, eax
0000674B 7468                   <1>         jz      short dmem_acc_6   ; Change PTE
0000674D D1E8                   <1>         shr     eax, 1       ; swap disk block (8 sectors) address
                                <1>         ; unlink swap disk block
0000674F E80CFBFFFF             <1>         call    unlink_swap_block
00006754 EB5F                   <1>         jmp     short dmem_acc_6
                                <1>
```

```
                                <1> dmem_acc_2:
00006756 A802                   <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                                <1>                              ; (must be 1)
00006758 7550                   <1>       jnz    short dmem_acc_4
                                <1>       ; Read only -duplicated- page (belongs to a parent or a child)
0000675A 66A90002               <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                                <1>                              ; as child's page ?
0000675E 7455                   <1>       jz     short dmem_acc_5 ; Change PTE but don't deallocate the page!
                                <1>
                                <1>       ;push  edi
                                <1>       ;push  esi
                                <1>
00006760 51                     <1>       push   ecx
                                <1>       ;push  ebx
00006761 8B1D[BC030300]         <1>       mov    ebx, [u.ppgdir] ; parent's page dir address (physical)
                                <1>
                                <1>       ; check the parent's PTE value is read only & same page or not..
00006767 89EF                   <1>       mov    edi, ebp
00006769 C1EF16                 <1>       shr    edi, PAGE_D_SHIFT ; 22
                                <1>       ; EDI = page directory entry index (0-1023)
0000676C 89EE                   <1>       mov    esi, ebp
0000676E C1EE0C                 <1>       shr    esi, PAGE_SHIFT ; 12
00006771 81E6FF030000           <1>       and    esi, PTE_MASK
                                <1>       ; ESI = page table entry index (0-1023)
                                <1>
00006777 66C1E702               <1>       shl    di, 2 ; * 4
0000677B 01FB                   <1>       add    ebx, edi ; PDE offset (for the parent)
0000677D 8B0F                   <1>       mov    ecx, [edi]
0000677F F6C101                 <1>       test   cl, PDE_A_PRESENT ; present (valid) or not ?
00006782 7425                   <1>       jz     short dmem_acc_3    ; parent process does not use this page
00006784 6681E100F0             <1>       and    cx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
00006789 66C1E602               <1>       shl    si, 2 ; *4
0000678D 01CE                   <1>       add    esi, ecx ; PTE offset (for the parent)
0000678F 8B1E                   <1>       mov    ebx, [esi]
00006791 F6C301                 <1>       test   bl, PTE_A_PRESENT ; present or not ?
00006794 7413                   <1>       jz     short dmem_acc_3    ; parent process does not use this page
00006796 662500F0               <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
0000679A 6681E300F0             <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
0000679F 39D8                   <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
000067A1 7506                   <1>       jne    short dmem_acc_3    ; not same page
                                <1>                              ; deallocate the child's page
000067A3 800E02                 <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
                                <1>       ;pop   ebx
000067A6 59                     <1>       pop    ecx
000067A7 EB0C                   <1>       jmp    short dmem_acc_5
                                <1> dmem_acc_3:
                                <1>       ;pop   ebx
000067A9 59                     <1>       pop    ecx
                                <1> dmem_acc_4:
000067AA 66A90004               <1>       test   ax, PTE_SHARED ; shared or direct memory access indicator
000067AE 7505                   <1>       jnz    short dmem_acc_5   ; AVL bit 1 = 1, do not deallocate this page!
                                <1>       ;
                                <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
000067B0 E804F6FFFF             <1>       call   deallocate_page
                                <1> dmem_acc_5:
                                <1>       ;pop   esi
                                <1>       ;pop   edi
                                <1> dmem_acc_6:
000067B5 89E8                   <1>       mov    eax, ebp ; physical page (offset=0) address
                                <1>       ; EAX = memory page address
                                <1>       ; EDX = PTE entry address (physical)
000067B7 660D0704               <1>       or     ax, PTE_A_PRESENT+PTE_A_USER+PTE_A_WRITE+PTE_SHARED
                                <1>                    ; present flag, bit 0 = 1
                                <1>                    ; user flag, bit 2 = 1
                                <1>                    ; writable flag, bit 1 = 1
                                <1>                    ; direct memory access flag, bit 10 = 1
                                <1>                    ; (This page must not be deallocated!)
000067BB 8902                   <1>       mov    [edx], eax  ; Update PTE value
000067BD 49                     <1>       dec    ecx ; remain count of contiguous pages
000067BE 741E                   <1>       jz     short dmem_acc_8
000067C0 81C500100000           <1>       add    ebp, PAGE_SIZE ; next physical page address
                                <1>       ; 22/07/2017
                                <1>       ;mov   eax, ebp
                                <1>       ; 12/05/2017
000067C6 8B1D[D4A00100]         <1>       mov    ebx, [base_addr] ; linear address (virtual+CORE)
000067CC 81C300100000           <1>       add    ebx, PAGE_SIZE      ; next linear address
000067D2 E938FFFFFF             <1>         jmp     dmem_acc_0
                                <1> dmem_acc_7:  ; ERROR !
000067D7 C7042404000000         <1>       mov    dword [esp], ERR_MINOR_IM
                                <1>              ; Insufficient memory (minor) error!
                                <1>              ; Major error = 0 (No protection fault)
                                <1>       ; cf = 1
                                <1> dmem_acc_8:
000067DE 58                     <1>       pop    eax
                                <1>       ;pop   edx
                                <1>       ;pop   ecx
                                <1>       ;pop   ebx
                                <1>       ;pop   ebp
000067DF C3                     <1>       retn
                                <1>
                                <1> deallocate_user_pages:
                                <1>       ; 20/05/2017
                                <1>       ; 15/05/2017
                                <1>       ; 20/02/2017
                                <1>       ; 19/02/2017 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;
                                <1>       ; Deallocate virtually contiguous user pages (memory block)
                                <1>       ; (caller: 'sysdalloc' system call)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      EBX = VIRTUAL ADDRESS (beginning address)
                                <1>       ;      ECX = byte count
                                <1>       ;      [u.pgdir] = user's page directory
                                <1>       ;      [u.ppdir] = parent's page directory
                                <1>       ;
```

```
                                <1>       ; OUTPUT ->
                                <1>       ;    If CF = 0
                                <1>       ;      EAX = Deallocated memory bytes
                                <1>       ;        (Even if shared or read only pages will not be
                                <1>       ;         deallocated on M.A.T., this byte count will be
                                <1>       ;         returned as virtually deallocated bytes; in fact
                                <1>       ;         virtually deallocated user pages * 4096.)
                                <1>       ;      EBX = Virtual address (as rounded up)
                                <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>       ;
000067E0 89DE                  <1>       mov    esi, ebx
000067E2 89F7                  <1>       mov    edi, esi
000067E4 01CF                  <1>       add    edi, ecx
000067E6 81C6FF0F0000          <1>       add    esi, PAGE_SIZE - 1  ; 4095 (round up)
000067EC C1EE0C                <1>       shr    esi, PAGE_SHIFT
000067EF C1EF0C                <1>       shr    edi, PAGE_SHIFT
000067F2 89F8                  <1>       mov    eax, edi ; end page
000067F4 29F0                  <1>       sub    eax, esi ; end page - start page
000067F6 0F86D5000000          <1>       jna    da_u_pd_err  ; < 1
000067FC 89F3                  <1>       mov    ebx, esi
000067FE C1E30C                <1>       shl    ebx, PAGE_SHIFT ; virtual address (as rounded up)
00006801 53                    <1>       push   ebx ; *
00006802 89C1                  <1>       mov    ecx, eax ; page count
00006804 C1E00C                <1>       shl    eax, PAGE_SHIFT ; byte count as adjusted
00006807 50                    <1>       push   eax ; **
00006808 8B1D[B8030300]        <1>       mov    ebx, [u.pgdir] ; physical addr of user's page dir
0000680E 81C600040000          <1>       add    esi, CORE/PAGE_SIZE
00006814 89F7                  <1>       mov    edi, esi
00006816 81E7FF030000          <1>       and    edi, PTE_MASK ; PTE entry in the page table
0000681C 57                    <1>       push   edi ; *** ; PTE index (of page directory)
0000681D C1EE0A                <1>       shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
00006820 89F2                  <1>       mov    edx, esi
                                <1>       ; EDX = PDE index
00006822 C1E602                <1>       shl    esi, 2 ; convert PDE index to dword offset
00006825 01DE                  <1>       add    esi, ebx ; add page directory address
                                <1> da_u_pd_1:
00006827 AD                    <1>       lodsd
                                <1>       ;
00006828 89F5                  <1>       mov    ebp, esi ; 20/02/2017
                                <1>       ; EBP = next PDE address
                                <1>       ;
0000682A A801                  <1>       test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
0000682C 0F8494000000          <1>       jz     da_u_pd_3 ; 20/05/2017
00006832 662500F0              <1>       and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
                                <1>       ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
00006836 8B3C24                <1>       mov    edi, [esp] ; ***
                                <1>       ; EDI = PTE index (of complete page directory)
                                <1>       ;and   edi, PTE_MASK ; PTE entry in the page table
00006839 C1E702                <1>       shl    edi, 2 ; convert PTE index to dword offset
0000683C 89FE                  <1>       mov    esi, edi ; PTE offset in page table (0-4092)
0000683E 01C6                  <1>       add    esi, eax ; now, esi points to requested PTE
                                <1> da_u_pt_0:
00006840 AD                    <1>       lodsd
00006841 A801                  <1>       test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
00006843 743F                  <1>       jz     short da_u_pt_1
                                <1>       ;
00006845 A802                  <1>       test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
                                <1>                                ; (must be 1)
00006847 7549                  <1>       jnz    short da_u_pt_3
                                <1>       ; Read only -duplicated- page (belongs to a parent or a child)
00006849 66A90002              <1>         test    ax, PTE_DUPLICATED ; Was this page duplicated
                                <1>                                    ; as child's page ?
0000684D 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)
0000684F 52                    <1>       push   edx ; ****
                                <1>       ; EDI = page table entry offset (0-4092)
00006850 8B1D[BC030300]        <1>       mov    ebx, [u.ppgdir] ; page directory of the parent process
00006856 66C1E202              <1>       shl    dx, 2 ; *4
0000685A 01D3                  <1>       add    ebx, edx ; PDE address (for the parent)
0000685C 8B13                  <1>       mov    edx, [ebx] ; page table address
0000685E F6C201                <1>       test   dl, PDE_A_PRESENT ; present (valid) or not ?
00006861 742E                  <1>       jz     short da_u_pt_2     ; parent process does not use this page
00006863 6681E200F0            <1>       and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
                                <1>       ; EDI = page table entry offset (0-4092)
00006868 01D7                  <1>       add    edi, edx     ; PTE address (for the parent)
0000686A 8B1F                  <1>       mov    ebx, [edi]
0000686C F6C301                <1>       test   bl, PTE_A_PRESENT ; present or not ?
0000686F 7420                  <1>       jz     short da_u_pt_2     ; parent process does not use this page
00006871 662500F0              <1>       and    ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
00006875 6681E300F0            <1>       and    bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
0000687A 39D8                  <1>       cmp    eax, ebx     ; parent's and child's pages are same ?
0000687C 7513                  <1>       jne    short da_u_pt_2     ; not same page
                                <1>                                ; deallocate the child's page
0000687E 800F02                <1>         or     byte [edi], PTE_A_WRITE ; convert to writable page (parent)
00006881 5A                    <1>       pop    edx ; ****
00006882 EB19                  <1>       jmp    short da_u_pt_4
                                <1> da_u_pt_1:
00006884 09C0                  <1>       or     eax, eax     ; swapped page ?
00006886 741C                  <1>       jz     short da_u_pt_5     ; no
                                <1>                                ; yes
00006888 D1E8                  <1>       shr    eax, 1
0000688A E8D1F9FFFF            <1>       call   unlink_swap_block ; Deallocate swapped page block
                                <1>                                ; on the swap disk (or in file)
0000688F EB13                  <1>       jmp    short da_u_pt_5
                                <1> da_u_pt_2:
00006891 5A                    <1>       pop    edx ; ****
                                <1> da_u_pt_3:
```

```
3389 00006892 66A90004            <1>       test   ax, PTE_SHARED      ; shared or direct memory access indicator
3390 00006896 7505                <1>       jnz    short da_u_pt_4     ; AVL bit 1 = 1, do not deallocate this page!
3391                              <1>       ;
3392                              <1>       ;and   ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3393 00006898 E81CF5FFFF          <1>       call   deallocate_page ; set the mem allocation bit of this page
3394                              <1> da_u_pt_4:
3395 0000689D C746FC00000000      <1>       mov    dword [esi-4], 0 ; clear/reset PTE (child, dupl. as parent)
3396                              <1> da_u_pt_5:
3397                              <1>       ; 20/05/2017
3398 000068A4 58                  <1>       pop    eax ; *** PTE index (of page directory)
3399 000068A5 49                  <1>       dec    ecx ; remain page count
3400 000068A6 7426                <1>       jz     short da_u_pd_4
3401 000068A8 40                  <1>       inc    eax ; next PTE
3402 000068A9 6625FF03            <1>       and    ax, PTE_MASK ; PTE entry index in the page table
3403 000068AD 50                  <1>       push   eax ; *** (save again)
3404                              <1>       ;mov   edi, eax
3405                              <1>       ;and   di, PTE_MASK
3406                              <1>       ;cmp   edi, PAGE_SIZE / 4 ; 1024
3407                              <1>       ;jnb   short da_u_pd_2
3408 000068AE 89C7                <1>       mov    edi, eax
3409 000068B0 C1E702              <1>       shl    edi, 2 ; convert index to dword offset
3410                              <1>       ;test  ax, PTE_MASK ; 3FFh
3411 000068B3 09C0                <1>       or     eax, eax
3412 000068B5 7589                <1>       jnz    short da_u_pt_0 ; 1-1023
3413                              <1> da_u_pd_2:
3414 000068B7 42                  <1>       inc    edx
3415                              <1>       ; 20/05/2017
3416 000068B8 6681E2FF03          <1>       and    dx, PTE_MASK  ; 3FFh
3417 000068BD 740F                <1>       jz     short da_u_pd_4  ; 0 (1024)
3418                              <1>       ;cmp   edx, 1024
3419                              <1>       ;jnb   short da_u_pd_4
3420 000068BF 89EE                <1>       mov    esi, ebp ; 20/02/2017
3421 000068C1 E961FFFFFF          <1>       jmp    da_u_pd_1
3422                              <1> da_u_pd_3:
3423                              <1>       ; 15/05/2017 (empty page directory entry)
3424 000068C6 81E900040000        <1>       sub    ecx, 1024
3425 000068CC 77E9                <1>       ja     short da_u_pd_2 ; 20/05/2017
3426                              <1> da_u_pd_4:
3427 000068CE 58                  <1>       pop    eax ; **
3428 000068CF 5B                  <1>       pop    ebx ; *
3429 000068D0 C3                  <1>       retn
3430                              <1>
3431                              <1> da_u_pd_err:
3432 000068D1 31C0                <1>       xor    eax, eax
3433 000068D3 F9                  <1>       stc
3434 000068D4 C3                  <1>       retn
3435                              <1>
3436                              <1> allocate_user_pages:
3437                              <1>       ; 20/05/2017
3438                              <1>       ; 01/05/2017, 02/05/2017, 15/05/2017
3439                              <1>       ; 04/03/2017
3440                              <1>       ; 20/02/2017 (TRDOS 386 = TRDOS v2.0)
3441                              <1>       ;
3442                              <1>       ; Allocate physically contiguous user pages (memory block)
3443                              <1>       ; (caller: 'sysalloc' system call)
3444                              <1>       ;
3445                              <1>       ; Note: This procedure does not alloc a page's itself
3446                              <1>       ;     (page bit) on Memory Allocation Table.
3447                              <1>       ;     (allocate_memory_block is needed before this proc)
3448                              <1>       ;
3449                              <1>       ; INPUT ->
3450                              <1>       ;     EAX = PHYSICAL ADDRESS (beginning address)
3451                              <1>       ;     EBX = VIRTUAL ADDRESS (beginning address)
3452                              <1>       ;     ECX = byte count (>=4096)
3453                              <1>       ;     [u.pgdir] = user's page directory
3454                              <1>       ;
3455                              <1>       ;     Note: All addresses are (must be) already adjusted
3456                              <1>       ;     to page      borders, otherwise, lower 12bits of addresses
3457                              <1>       ;     and byte count would be truncated.
3458                              <1>       ;
3459                              <1>       ; OUTPUT ->
3460                              <1>       ;     none
3461                              <1>       ;
3462                              <1>       ;     CF = 1 -> insufficient memory error
3463                              <1>       ;
3464                              <1>       ; Note: All pages will be allocated in physical page order
3465                              <1>       ;     from the beginning page address.
3466                              <1>       ;     * A new page table will be added to the page dir
3467                              <1>       ;       when the requested PDE is invalid.
3468                              <1>       ;     * Those pages will not be added to swap queue
3469                              <1>       ;       because main purpose of this allocation is to
3470                              <1>       ;       set a direct memory access (DMA controller) buffer.
3471                              <1>       ;      (Swapping out a page in a DMA buffer would be wrong!)
3472                              <1>       ;     * Previous content of page tables (PTEs) would be
3473                              <1>       ;       (should be) deallocated before entering this
3474                              <1>       ;       procedure. So, new page table entries (PTEs)
3475                              <1>       ;       directly will be written without checking
3476                              <1>       ;       their previous content.
3477                              <1>       ;     * Only solution to increase free memory by removing
3478                              <1>       ;       that non-swappable memory block is to terminate
3479                              <1>       ;       the process or to wait until the process will
3480                              <1>       ;       deallocate that memory block as itself. ('sysdalloc')
3481                              <1>       ;       (No problem, if the process does not grab all of
3482                              <1>       ;       -very big amount of- free memory by using
3483                              <1>       ;       'sysalloc' system call!?)
3484                              <1>       ;       (Even if the process has grabbed all of free memory,
3485                              <1>       ;       no problem if the process is not running in
3486                              <1>       ;       multitasking mode. No problem in multitasking
3487                              <1>       ;       mode if there is not another process which is running
3488                              <1>       ;       or waiting or sleeping for an event as it's pages
3489                              <1>       ;       are swapped-out. But a new process can not start to
3490                              <1>       ;       run if all of free memory has beeen allocated
3491                              <1>       ;       by running processes. Deallocation -'sysdalloc'-
3492                              <1>       ;       or terminate a running process is needed
3493                              <1>       ;       in order to run a new process.)
```

```
3494                                    <1>      ;
3495                                    <1>      ; Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP
3496                                    <1>      ;
3497                                    <1>
3498                                    <1>      ; 01/05/2017
3499 000068D5 662500F0                  <1>      and    ax, ~PAGE_OFF
3500 000068D9 6681E300F0                <1>      and    bx, ~PAGE_OFF
3501                                    <1>      ; 02/05/2017
3502 000068DE BD00F0FFFF                <1>      mov    ebp, 0FFFFF000h ; 4 Giga Bytes - 4096 Bytes (for Stack)
3503 000068E3 C1E90C                    <1>      shr    ecx, PAGE_SHIFT ; page count
3504 000068E6 83F901                    <1>      cmp    ecx, 1
3505 000068E9 7251                      <1>      jb     short a_u_im_retn
3506 000068EB 89C2                      <1>      mov    edx, eax
3507 000068ED 01CA                      <1>      add    edx, ecx
3508 000068EF 724B                      <1>      jc     short a_u_im_retn
3509 000068F1 39D5                      <1>      cmp    ebp, edx
3510 000068F3 7247                      <1>      jb     short a_u_im_retn
3511 000068F5 89DA                      <1>      mov    edx, ebx
3512 000068F7 81C200004000              <1>      add    edx, CORE
3513 000068FD 723D                      <1>      jc     short a_u_im_retn
3514 000068FF 01CA                      <1>      add    edx, ecx
3515 00006901 7239                      <1>      jc     short a_u_im_retn
3516 00006903 39D5                      <1>      cmp    ebp, edx
3517 00006905 7235                      <1>      jb     short a_u_im_retn
3518                                    <1>      ;
3519 00006907 89C5                      <1>      mov    ebp, eax ; physical address
3520 00006909 89DE                      <1>      mov    esi, ebx
3521 0000690B 81C600004000              <1>      add    esi, CORE ; start of user's memory (4M)
3522 00006911 C1EE0C                    <1>      shr    esi, PAGE_SHIFT ; higher 20 bits of the linear address
3523                                    <1>      ;shr   ecx, PAGE_SHIFT ; page count
3524 00006914 8B1D[B8030300]            <1>      mov    ebx, [u.pgdir] ; physical addr of user's page dir
3525 0000691A 89F7                      <1>      mov    edi, esi
3526 0000691C 81E7FF030000              <1>      and    edi, PTE_MASK ; PTE entry index in the page table
3527 00006922 57                        <1>      push   edi  ; * ; PTE index (in page directory)
3528 00006923 C1EE0A                    <1>      shr    esi, PAGE_D_SHIFT - PAGE_SHIFT ; 22-12=10
3529 00006926 89F2                      <1>      mov    edx, esi
3530                                    <1>      ; EDX = PDE index
3531 00006928 C1E602                    <1>      shl    esi, 2 ; convert PDE index to dword offset
3532 0000692B 01DE                      <1>      add    esi, ebx ; add page directory address
3533                                    <1> a_u_pd_0:
3534 0000692D AD                        <1>      lodsd
3535                                    <1>      ;
3536 0000692E 89F3                      <1>      mov    ebx, esi ; next PDE address
3537                                    <1>      ;
3538 00006930 A801                      <1>      test   al, PDE_A_PRESENT ; bit 0, present flag (must be 1)
3539 00006932 7513                      <1>      jnz    short a_u_pd_2
3540                                    <1>      ;
3541                                    <1>      ; empty PDE (it does not point to valid page table address)
3542 00006934 E8A2F2FFFF                <1>      call   allocate_page  ; (allocate a new page table)
3543 00006939 7302                      <1>      jnc    short a_u_pd_1 ; OK... now, we have a new page table.
3544                                    <1>      ; cf = 1
3545                                    <1>      ; There is not a free memory page to allocate a new page table !!!
3546 0000693B 5E                        <1>      pop    esi ; *
3547                                    <1> a_u_im_retn:
3548 0000693C C3                        <1>      retn   ; return to 'sysalloc' with 'insufficient memory' error
3549                                    <1>      ;
3550                                    <1> a_u_pd_1: ; clear the new page table content
3551                                    <1>      ; EAX = Physical (base) address of the new page table
3552 0000693D E813F3FFFF                <1>      call   clear_page ; Clear page content
3553                                    <1>      ;
3554 00006942 0C07                      <1>      or     al, PDE_A_PRESENT + PDE_A_WRITE + PDE_A_USER
3555                                    <1>             ; set bit 0, bit 1 and bit 2 to 1
3556                                    <1>             ; (present, writable, user)
3557 00006944 8946FC                    <1>      mov    [esi-4], eax
3558                                    <1> a_u_pd_2:
3559 00006947 662500F0                  <1>      and    ax, PDE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3560                                    <1>      ; EAX = PHYSICAL (flat) ADDRESS OF THE PAGE TABLE
3561 0000694B 8B3C24                    <1>      mov    edi, [esp] ; *
3562                                    <1>      ; EDI = PTE index (of page directory)
3563                                    <1>      ;and   edi, PTE_MASK ; PTE entry index in the page table
3564                                    <1>      ; EBX = next PDE address
3565 0000694E 89FE                      <1>      mov    esi, edi ; PTE index in page table (0-1023)
3566 00006950 C1E702                    <1>      shl    edi, 2 ; convert PTE index to dword offset
3567 00006953 01C7                      <1>      add    edi, eax ; now, edi points to requested PTE
3568                                    <1> a_u_pt_0:
3569                                    <1>      ; 02/05/2017
3570 00006955 8B07                      <1>      mov    eax, [edi]
3571                                    <1>      ;
3572 00006957 A801                      <1>      test   al, PTE_A_PRESENT ; bit 0, present flag (must be 1)
3573 00006959 7445                      <1>      jz     short a_u_pt_1
3574                                    <1>      ;
3575 0000695B A802                      <1>      test   al, PTE_A_WRITE   ; bit 1, writable (r/w) flag
3576                                    <1>                               ; (must be 1)
3577 0000695D 7550                      <1>      jnz    short a_u_pt_3
3578                                    <1>      ; Read only -duplicated- page (belongs to a parent or a child)
3579 0000695F 66A90002                  <1>        test    ax, PTE_DUPLICATED ; Was this page duplicated
3580                                    <1>                                 ; as child's page ?
3581 00006963 7455                      <1>      jz     short a_u_pt_4      ; Clear PTE but don't deallocate the page!
3582                                    <1>      ;
3583                                    <1>      ; check the parent's PTE value is read only & same page or not..
3584                                    <1>      ; EDX = page directory entry index (0-1023)
3585 00006965 52                        <1>      push   edx ; **
3586 00006966 53                        <1>      push   ebx ; ***
3587                                    <1>      ; ESI = page table entry index (0-1023)
3588                                    <1>      ;push  esi ; **** ; 20/05/2017
3589 00006967 8B1D[BC030300]            <1>      mov    ebx, [u.ppgdir] ; page directory of the parent process
3590 0000696D 66C1E202                  <1>      shl    dx, 2 ; *4
3591 00006971 01D3                      <1>      add    ebx, edx ; PTE address,0 (for the parent)
3592 00006973 8B13                      <1>      mov    edx, [ebx] ; page table address
3593 00006975 F6C201                    <1>      test   dl, PDE_A_PRESENT ; present (valid) or not ?
3594 00006978 7433                      <1>      jz     short a_u_pt_2      ; parent process does not use this page
3595 0000697A 6681E200F0                <1>      and    dx, PDE_A_CLEAR ; 0F000h ; Clear attribute bits
3596 0000697F 66C1E602                  <1>      shl    si, 2 ; *4
3597                                    <1>      ; ESI = page table entry offset (0-4092)
3598 00006983 01D6                      <1>      add    esi, edx     ; PTE address (for the parent)
```

```
3599 00006985 8B1E              <1>       mov   ebx, [esi]
3600 00006987 F6C301            <1>       test  bl, PTE_A_PRESENT ; present or not ?
3601 0000698A 7421              <1>       jz    short a_u_pt_2      ; parent process does not use this page
3602 0000698C 662500F0          <1>       and   ax, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3603 00006990 6681E300F0        <1>       and   bx, PTE_A_CLEAR ; 0F000h ; Clear attribute bits
3604 00006995 39D8              <1>       cmp   eax, ebx      ; parent's and child's pages are same ?
3605 00006997 7514              <1>       jne   short a_u_pt_2      ; not same page
3606                            <1>                             ; deallocate the child's page
3607 00006999 800E02            <1>         or      byte [esi], PTE_A_WRITE ; convert to writable page (parent)
3608                            <1>       ;pop  esi ; **** ; 20/05/2017
3609 0000699C 5B                <1>       pop   ebx ; ***
3610 0000699D 5A                <1>       pop   edx ; **
3611 0000699E EB1A              <1>       jmp   short a_u_pt_4
3612                            <1> a_u_pt_1:
3613 000069A0 09C0              <1>       or    eax, eax      ; swapped page ?
3614 000069A2 7416              <1>       jz    short a_u_pt_4      ; no
3615                            <1>                             ; yes
3616 000069A4 D1E8              <1>       shr   eax, 1
3617 000069A6 E8B5F8FFFF        <1>       call  unlink_swap_block ; Deallocate swapped page block
3618                            <1>                               ; on the swap disk (or in file)
3619 000069AB EB0D              <1>       jmp   short a_u_pt_4
3620                            <1> a_u_pt_2:
3621                            <1>       ;pop  esi ; **** ; 20/05/2017
3622 000069AD 5B                <1>       pop   ebx ; ***
3623 000069AE 5A                <1>       pop   edx ; **
3624                            <1> a_u_pt_3:
3625 000069AF 66A90004          <1>       test  ax, PTE_SHARED      ; shared or direct memory access indicator
3626 000069B3 7505              <1>       jnz   short a_u_pt_4      ; AVL bit 1 = 1, do not deallocate this page!
3627                            <1>       ;
3628                            <1>       ;and  ax, PTE_A_CLEAR ; 0F000h ; clear lower 12 (attribute) bits
3629 000069B5 E8FFF3FFFF        <1>       call  deallocate_page ; set the mem allocation bit of this page
3630                            <1>       ;
3631                            <1> a_u_pt_4:
3632 000069BA 89E8              <1>       mov   eax, ebp ; physical address
3633 000069BC 0C07              <1>       or    al, PTE_A_PRESENT + PTE_A_WRITE + PTE_A_USER ; 04/03/2017
3634 000069BE AB                <1>       stosd
3635 000069BF 5E                <1>       pop   esi ; * ; 20/05/2017
3636 000069C0 49                <1>       dec   ecx ; remain page count
3637 000069C1 7417              <1>       jz    short a_u_pd_5
3638 000069C3 81C500100000      <1>       add   ebp, PAGE_SIZE
3639 000069C9 46                <1>       inc   esi ; next PTE (index)
3640                            <1>       ; 20/05/2017
3641                            <1>       ;cmp  esi, PAGE_SIZE/4 ; 1024
3642                            <1>       ;jb   short a_u_pt_0
3643 000069CA 6681E6FF03        <1>       and   si, PTE_MASK ; 3FFh (0 to 1023)
3644 000069CF 56                <1>       push  esi ; *
3645 000069D0 7583              <1>       jnz   short a_u_pt_0 ; > 0 (<1024)
3646                            <1> a_u_pd_3:
3647 000069D2 42                <1>       inc   edx
3648                            <1> ;     cmp   edx, 1024
3649                            <1> ;     jnb   short a_u_pd_4 ; 02/05/2017 (error!, ecx > 0)
3650 000069D3 89DE              <1>       mov   esi, ebx ; the next PDE address
3651 000069D5 E953FFFFFF        <1>       jmp   a_u_pd_0
3652                            <1> a_u_pd_4:
3653                            <1>       ; 02/05/2017
3654                            <1> ;     stc
3655                            <1> a_u_pd_5:
3656                            <1>       ; 20/05/2017
3657                            <1>       ;pop  edi ; *
3658 000069DA C3                <1>       retn
3659                            <1>
3660                            <1> allocate_lfb_pages_for_kernel:
3661                            <1>       ; 15/12/2020
3662                            <1>       ; 14/12/2020 - TRDOS 386 v2.0.3
3663                            <1>       ; Set kernel page tables for linear frame buffer
3664                            <1>       ; (this procedure will be called by kernel only)
3665                            <1>       ;
3666                            <1>       ; Input:
3667                            <1>       ;     [LFB_ADDR] = linear frame buffer base address
3668                            <1>       ;     [LFB_SIZE] = linear frame buffer size in bytes
3669                            <1>       ; Output:
3670                            <1>       ;     none
3671                            <1>       ;     cf = 1 -> error
3672                            <1>       ;
3673                            <1>       ; Modified registers: eax, ecx, edx, edi
3674                            <1>
3675 000069DB 8B3D[2C120300]    <1>       mov   edi, [LFB_ADDR]
3676 000069E1 8B15[30120300]    <1>       mov   edx, [LFB_SIZE]
3677                            <1>
3678 000069E7 C1EF16            <1>       shr   edi, 22      ; convert address to page number
3679                            <1>                          ; and then convert it to PDE entry offset
3680                            <1>                          ; (1 PDE is for 4MB, 22 bit shift)
3681                            <1>
3682 000069EA 66C1E702          <1>       shl   di, 2  ; * 4 for offset
3683                            <1>
3684                            <1>       ;add  edx, 4095
3685 000069EE C1EA0C            <1>       shr   edx, 12      ; convert LFB size to LFB page count
3686                            <1>
3687 000069F1 89D1              <1>       mov   ecx, edx ; * ; LFB page count
3688                            <1>
3689 000069F3 81C1FF030000      <1>       add   ecx, 1023 ; page count + 1023
3690 000069F9 C1E90A            <1>       shr   ecx, 10 ; convert to page directory entry count
3691                            <1>                     ; (page table count)
3692 000069FC 51                <1>       push  ecx ; **
3693 000069FD C1E10C            <1>       shl   ecx, 12 ; convert to byte count
3694                            <1>
3695 00006A00 31C0              <1>       xor   eax, eax ; first available pages
3696                            <1>
3697                            <1>       ; allocate contiguous memory block for these kernel pages
3698                            <1>
3699 00006A02 E87EFAFFFF        <1>       call  allocate_memory_block
3700                            <1>       ; eax = start address of (contiguous) memory block
3701 00006A07 59                <1>       pop   ecx ; ** ; PDE count
3702 00006A08 7301              <1>       jnc   short a_lfb_k_1
3703                            <1>       ; error (cf=1)
```

```
3704 00006A0A C3                <1>       retn
3705                            <1> a_lfb_k_1:
3706                            <1>       ; Allocate (new) page tables in kernel's page directory
3707 00006A0B 51                <1>       push   ecx ; PDE (page table) count
3708 00006A0C 50                <1>       push   eax ; start address of contiguous memory pages
3709                            <1>                  ; (at page boundary)
3710                            <1>       ; edi = 1st page directory entry offset
3711 00006A0D 033D[188A0100]    <1>       add    edi, [k_page_dir] ; Kernel's Page Dir Address
3712                            <1> a_lfb_k_2:
3713 00006A13 660D0304          <1>       or     ax, PDE_A_PRESENT + PDE_A_WRITE + PDE_EXTERNAL
3714                            <1>                           ; supervisor + read&write + present
3715                            <1>                           ; + external memory block (LFB)
3716 00006A17 AB                <1>       stosd
3717 00006A18 0500100000        <1>       add    eax, 4096
3718 00006A1D E2F4              <1>       loop   a_lfb_k_2
3719                            <1>
3720 00006A1F 5F                <1>       pop    edi ; start addr of contiguous memory pages
3721 00006A20 59                <1>       pop    ecx ; page table (PDE) count
3722                            <1>
3723                            <1>       ; Allocate pages in (new) kernel page tables
3724                            <1>
3725                            <1>       ; (Note: page tables are contiguous in pyhsical memory)
3726 00006A21 C1E10A            <1>       shl    ecx, 10 ; * 1024, convert to (total) PTE count
3727                            <1>
3728 00006A24 A1[2C120300]      <1>       mov    eax, [LFB_ADDR]
3729                            <1>       ; edx = LFB page count
3730                            <1>       ;and   ax, ~4095  ; lw of LFB address is 0
3731                            <1> a_lfb_k_3:
3732 00006A29 660D0304          <1>       or     ax, PTE_A_PRESENT + PTE_A_WRITE + PTE_EXTERNAL
3733                            <1>                           ; supervisor + read&write + present
3734                            <1>                           ; + external memory block (LFB)
3735 00006A2D AB                <1>       stosd
3736 00006A2E 4A                <1>       dec    edx
3737 00006A2F 7408              <1>       jz     short a_lfb_k_4 ; LFB size has been completed (!?)
3738 00006A31 0500100000        <1>       add    eax, 4096
3739 00006A36 E2F1              <1>       loop   a_lfb_k_3
3740                            <1>
3741 00006A38 C3                <1>       retn
3742                            <1>
3743                            <1> a_lfb_k_4:
3744                            <1>       ; clear PTEs for empty/free pages
3745                            <1>       ;      (if there are after LFB !?)
3746 00006A39 31C0              <1>       xor    eax, eax ; clear page table entry (empty)
3747 00006A3B F3AB              <1>       rep    stosd
3748 00006A3D C3                <1>       retn
3749                            <1>
3750                            <1> ;deallocate_lfb_pages_for_kernel:
3751                            <1>       ; 15/12/2020
3752                            <1>       ; 14/12/2020 - TRDOS 386 v2.0.3
3753                            <1>       ; Reset/Release kernel page tables
3754                            <1>       ;       which are used for linear frame buffer
3755                            <1>       ; (this procedure will be called by kernel only)
3756                            <1>       ;
3757                            <1>       ; Input:
3758                            <1>       ;      [LFB_ADDR] = linear frame buffer base address
3759                            <1>       ;      [FFB_SIZE] = linear frame buffer size in bytes
3760                            <1>       ; Output:
3761                            <1>       ;      none
3762                            <1>       ;
3763                            <1>       ; Modified registers: eax, ecx, edi
3764                            <1>
3765                            <1>       ;mov   edi, [LFB_ADDR]
3766                            <1>       ;mov   ecx, [LFB_SIZE]
3767                            <1>       ;
3768                            <1>       ;shr   edi, 22      ; convert address to page number
3769                            <1>       ;            ; and then convert it to PDE entry offset
3770                            <1>       ;            ; (1 PDE is for 4MB, 22 bit shift)
3771                            <1>       ;
3772                            <1>       ;shl   di, 2  ; * 4 for offset
3773                            <1>       ;
3774                            <1>       ;;add  ecx, 4095
3775                            <1>       ;shr   ecx, 12      ; convert LFB size to page count
3776                            <1>       ;
3777                            <1>       ;add   ecx, 1023 ; page count + 1023
3778                            <1>       ;shr   ecx, 10 ; convert to page directory entry count
3779                            <1>       ;            ; (page table count)
3780                            <1>       ;push  ecx ; *
3781                            <1>       ;shl   ecx, 12 ; convert to byte count
3782                            <1>       ;
3783                            <1>       ;xor   eax, eax ; first available pages
3784                            <1>       ;
3785                            <1>       ;; deallocate contiguous memory block for kernel pages
3786                            <1>       ;
3787                            <1>       ;call  deallocate_memory_block
3788                            <1>       ;
3789                            <1>       ;pop   ecx ; * ; PDE count
3790                            <1>       ;
3791                            <1>       ;; Release/Free PDEs (page tables) in kernel's page dir
3792                            <1>       ;; edi = 1st page directory entry offset
3793                            <1>       ;add   edi, [k_page_dir] ; Kernel's Page Dir Address
3794                            <1>       ;sub   eax, eax ; clear (also invalidate)
3795                            <1>       ;rep   stosd
3796                            <1>       ;
3797                            <1>       ;retn
3798                            <1>
3799                            <1> ; /// End Of MEMORY MANAGEMENT FUNCTIONS ///
3800                            <1>
3801                            <1> ;; Data:
3802                            <1>
3803                            <1> ; 09/03/2015
3804                            <1> ;swpq_count: dw 0 ; count of pages on the swap que
3805                            <1> ;swp_drv:    dd 0 ; logical drive description table address of the swap drive/disk
3806                            <1> ;swpd_size:  dd 0 ; size of swap drive/disk (volume) in sectors (512 bytes).

3807                            <1> ;swpd_free:  dd 0 ; free page blocks (4096 bytes) on swap disk/drive (logical)
```

```
3808                                    <1> ;swpd_next:  dd 0 ; next free page block
3809                                    <1> ;swpd_last:  dd 0 ; last swap page block
2896                                        %include 'timer.s'   ; 17/01/2015
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - timer.s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 15/01/2017
   5                                    <1> ; ----------------------------------------------------------------------------
   6                                    <1> ; Beginning: 17/01/2016
   7                                    <1> ; ----------------------------------------------------------------------------
   8                                    <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                                    <1> ; ----------------------------------------------------------------------------
  10                                    <1> ; Turkish Rational DOS
  11                                    <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                                    <1> ;
  13                                    <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  14                                    <1> ;
  15                                    <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
  16                                    <1> ; ****************************************************************************
  17                                    <1>
  18                                    <1> ; TRDOS 386  (TRDOS v2.0) Kernel - TIMER & REAL TIME CLOCK (BIOS) FUNCTIONS
  19                                    <1>
  20                                    <1> ; IBM PC-AT BIOS Source Code ('BIOS2.ASM')
  21                                    <1> ; TITLE BIOS2 ---- 06/10/85 BIOS INTERRUPT ROUTINES
  22                                    <1>
  23                                    <1> ;
  24                                    <1> ; ///////// TIMER (& REAL TIME CLOCK) FUNCTIONS ///////////////
  25                                    <1>
  26                                    <1> int1Ah:
  27                                    <1>       ; 29/01/2016
  28                                    <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
  29 00006A3E 9C                        <1>       pushfd
  30 00006A3F 0E                        <1>       push   cs
  31 00006A40 E801000000                <1>       call   TIME_OF_DAY_1
  32 00006A45 C3                        <1>       retn
  33                                    <1>
  34                                    <1> ;--- INT  1A H -- (TIME OF DAY) ------------------------------------------------
  35                                    <1> ;       THIS BIOS ROUTINE ALLOWS THE CLOCKS TO BE SET OR READ                :
  36                                    <1> ;                                                                            :
  37                                    <1> ; PARAMETERS:                                                                :
  38                                    <1> ;     (AH) = 00H  READ THE CURRENT SETTING AND RETURN WITH,                  :
  39                                    <1> ;                       (CX) = HIGH PORTION OF COUNT                         :
  40                                    <1> ;                       (DX) = LOW PORTION OF COUNT                          :
  41                                    <1> ;                       (AL) = 0 TIMER HAS NOT PASSED 24 HOURS SINCE LAST READ :
  42                                    <1> ;                              1 IF ON ANOTHER DAY. (RESET TO ZERO AFTER READ) :
  43                                    <1> ;                                                                            :
  44                                    <1> ;     (AH) = 01H  SET THE CURRENT CLOCK USING,                               :
  45                                    <1> ;                  (CX) = HIGH PORTION OF COUNT                              :
  46                                    <1> ;                  (DX) = LOW PORTION OF COUNT.                              :
  47                                    <1> ;                                                                            :
  48                                    <1> ;                NOTE: COUNTS OCCUR AT THE RATE OF 1193180/65536 COUNTS/SECOND :
  49                                    <1> ;                             (OR ABOUT 18.2 PER SECOND -- SEE EQUATES)      :
  50                                    <1> ;                                                                            :
  51                                    <1> ;     (AH) = 02H  READ THE REAL TIME CLOCK AND RETURN WITH,                  :
  52                                    <1> ;                       (CH) = HOURS IN BCD (00-23)                          :
  53                                    <1> ;                       (CL) = MINUTES IN BCD (00-59)                        :
  54                                    <1> ;                       (DH) = SECONDS IN BCD (00-59)                        :
  55                                    <1> ;                       (DL) = DAYLIGHT SAVINGS ENABLE (00-01)               :
  56                                    <1> ;                                                                            :
  57                                    <1> ;     (AH) = 03H  SET THE REAL TIME CLOCK USING,                             :
  58                                    <1> ;                      (CH) = HOURS IN BCD (00-23)                           :
  59                                    <1> ;                      (CL) = MINUTES IN BCD (00-59)                         :
  60                                    <1> ;                      (DH) = SECONDS IN BCD (00-59)                         :
  61                                    <1> ;                      (DL) = 01 IF DAYLIGHT SAVINGS ENABLE OPTION, ELSE 00. :
  62                                    <1> ;                                                                            :
  63                                    <1> ;               NOTE: (DL) = 00 IF DAYLIGHT SAVINGS TIME ENABLE IS NOT ENABLED. :
  64                                    <1> ;                     (DL) = 01 ENABLES TWO SPECIAL UPDATES THE LAST SUNDAY IN  :
  65                                    <1> ;                  APRIL   (1:59:59 --> 3:00:00 AM) AND THE LAST SUNDAY IN     :
  66                                    <1> ;                     OCTOBER (1:59:59 --> 1:00:00 AM) THE FIRST TIME.         :
  67                                    <1> ;                                                                            :
  68                                    <1> ;     (AH) = 04H  READ THE DATE FROM THE REAL TIME CLOCK AND RETURN WITH,    :
  69                                    <1> ;                       (CH) = CENTURY IN BCD (19 OR 20)                     :
  70                                    <1> ;                       (CL) = YEAR IN BCD (00-99)                           :
  71                                    <1> ;                       (DH) = MONTH IN BCD (01-12)                          :
  72                                    <1> ;                       (DL) = DAY IN BCD (01-31).                           :
  73                                    <1> ;                                                                            :
  74                                    <1> ;     (AH) = 05H  SET THE DATE INTO THE REAL TIME CLOCK USING,               :
  75                                    <1> ;                      (CH) = CENTURY IN BCD (19 OR 20)                      :
  76                                    <1> ;                      (CL) = YEAR IN BCD (00-99)                            :
  77                                    <1> ;                      (DH) = MONTH IN BCD (01-12)                           :
  78                                    <1> ;                      (DL) = DAY IN BCD (01-31).                            :
  79                                    <1> ;                                                                            :
  80                                    <1> ;     (AH) = 06H  SET THE ALARM TO INTERRUPT AT SPECIFIED TIME,              :
  81                                    <1> ;                      (CH) = HOURS IN BCD (00-23 (OR FFH))                  :
  82                                    <1> ;                      (CL) = MINUTES IN BCD (00-59 (OR FFH))                :
  83                                    <1> ;                      (DH) = SECONDS IN BCD (00-59 (OR FFH))                :
  84                                    <1> ;                                                                            :
  85                                    <1> ;     (AH) = 07H  RESET THE ALARM INTERRUPT FUNCTION.                        :
  86                                    <1> ;                                                                            :
  87                                    <1> ; NOTES: FOR ALL RETURNS CY= 0 FOR SUCCESSFUL OPERATION.                     :
  88                                    <1> ;        FOR (AH)= 2, 4, 6 - CARRY FLAG SET IF REAL TIME CLOCK NOT OPERATING. :
  89                                    <1> ;        FOR (AH)= 6 - CARRY FLAG SET IF ALARM ALREADY ENABLED.              :
  90                                    <1> ;        FOR THE ALARM FUNCTION (AH = 6) THE USER MUST SUPPLY A ROUTINE AND  :
  91                                    <1> ;         INTERCEPT THE CORRECT ADDRESS IN THE VECTOR TABLE FOR INTERRUPT 4AH.     :
  92                                    <1> ;         USE 0FFH FOR ANY "DO NOT CARE" POSITION FOR INTERVAL INTERRUPTS.   :
  93                                    <1> ;        INTERRUPTS ARE DISABLED DURING DATA MODIFICATION.                   :
  94                                    <1> ;        AH & AL ARE RETURNED MODIFIED AND NOT DEFINED EXCEPT WHERE INDICATED.    :
  95                                    <1> ;----------------------------------------------------------------------------
  96                                    <1>
  97                                    <1> ; 15/01/2017
  98                                    <1> ; 14/01/2017
  99                                    <1> ; 07/01/2017
 100                                    <1> ; 02/01/2017
 101                                    <1> ; 29/05/2016
 102                                    <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
00006A46 80642408FE         <1>       and   byte [esp+8], 11111110b   ; clear carry bit of eflags register
                            <1>       ;
00006A4B 80FC08             <1>       cmp   ah, (RTC_TBE-RTC_TB)/4    ; CHECK IF COMMAND IN VALID RANGE (0-7)
00006A4E F5                 <1>       cmc                       ; COMPLEMENT CARRY FOR ERROR EXIT
                            <1>       ; (*) jc short TIME_9           ; EXIT WITH CARRY = 1 IF NOT VALID
00006A4F 721A               <1>       jc    short _TIME_9 ; 29/05/2016
                            <1>
00006A51 1E                 <1>       push  ds
00006A52 56                 <1>       push  esi
00006A53 66BE1000           <1>       mov   si, KDATA           ; kernel data segment
00006A57 8EDE               <1>       mov   ds, si
                            <1>
                            <1>       ;;15/01/2017
                            <1>       ; 14/01/2017
                            <1>       ; 02/01/2017
                            <1>       ;;mov  byte [intflg], 35h  ; date & time interrupt
                            <1>       ;sti
                            <1>       ;
00006A59 C0E402             <1>       shl   ah, 2               ; convert function to dword offset
00006A5C 0FB6F4             <1>       movzx esi, ah                   ; PLACE INTO ADDRESSING REGISTER
                            <1>       ;cli                      ; NO INTERRUPTS DURING TIME FUNCTIONS
00006A5F FF96[716A0000]     <1>       call  [esi+RTC_TB]        ; VECTOR TO FUNCTION REQUESTED WITH CY=0
                            <1>                                 ; RETURN WITH CARRY FLAG SET FOR RESULT
                            <1>       ;sti                      ; INTERRUPTS BACK ON
00006A65 B400               <1>       mov   ah, 0               ; CLEAR (AH) TO ZERO
00006A67 5E                 <1>       pop   esi                 ; RECOVER USERS REGISTER
00006A68 1F                 <1>       pop   ds                  ; RECOVER USERS SEGMENT SELECTOR
                            <1>
                            <1>       ;;15/01/2017
                            <1>       ; 02/01/2017
                            <1>       ;;mov  byte [ss:intflg], 0 ; 07/01/2017
                            <1>
                            <1> ;TIME_9:
                            <1>                                 ; RETURN WITH CY= 0 IF NO ERROR
                            <1>       ; (*) 29/05/2016
                            <1>       ; (*) retf 4 ; skip eflags on stack
00006A69 7305               <1>       jnc   short _TIME_10
                            <1> _TIME_9:
                            <1>       ; 29/05/2016 -set carry flag on stack-
                            <1>       ; [esp] = EIP
                            <1>       ; [esp+4] = CS
                            <1>       ; [esp+8] = E-FLAGS
00006A6B 804C240801         <1>       or    byte [esp+8], 1     ; set carry bit of eflags register
                            <1>       ; [esp+12] = ESP (user)
                            <1>       ; [esp+16] = SS (User)
                            <1> _TIME_10:
00006A70 CF                 <1>       iretd
                            <1>
                            <1>       ; (*) 29/05/2016 - 'ref 4' intruction causes to stack fault
                            <1>       ; (OUTER-PRIVILEGE-LEVEL)
                            <1>       ; INTEL 80386 PROGRAMMER'S REFERENCE MANUAL 1986
                            <1>       ; // RETF instruction:
                            <1>       ;
                            <1>       ; IF OperandMode=32 THEN
                            <1>       ;    Load CS:EIP from stack;
                            <1>       ;    Set CS RPL to CPL;
                            <1>       ;    Increment eSP by 8 plus the immediate offset if it exists;
                            <1>       ;    Load SS:eSP from stack;
                            <1>       ; ELSE (* OperandMode=16 *)
                            <1>       ;    Load CS:IP from stack;
                            <1>       ;    Set CS RPL to CPL;
                            <1>       ;    Increment eSP by 4 plus the immediate offset if it exists;
                            <1>       ;    Load SS:eSP from stack;
                            <1>       ; FI;
                            <1>       ;
                            <1>       ; //
                            <1>                                 ; ROUTINE VECTOR TABLE (AH)=
                            <1> RTC_TB:
00006A71 [916A0000]         <1>       dd    RTC_00              ; 0 = READ CURRENT CLOCK COUNT
00006A75 [A46A0000]         <1>       dd    RTC_10              ; 1 = SET CLOCK COUNT
00006A79 [B26A0000]         <1>       dd    RTC_20              ; 2 = READ THE REAL TIME CLOCK TIME
00006A7D [E16A0000]         <1>       dd    RTC_30              ; 3 = SET REAL TIME CLOCK TIME
00006A81 [236B0000]         <1>       dd    RTC_40              ; 4 = READ THE REAL TIME CLOCK DATE
00006A85 [506B0000]         <1>       dd    RTC_50              ; 5 = SET REAL TIME CLOCK DATE
00006A89 [9D6B0000]         <1>       dd    RTC_60              ; 6 = SET THE REAL TIME CLOCK ALARM
00006A8D [F06B0000]         <1>       dd    RTC_70              ; 7 = RESET ALARM
                            <1>
                            <1> RTC_TBE     equ   $
                            <1>
                            <1> RTC_00:                         ; READ TIME COUNT
00006A91 A0[9C8A0100]       <1>       mov   al, [TIMER_OFL]           ; GET THE OVERFLOW FLAG
00006A96 C605[9C8A0100]00   <1>       mov   byte [TIMER_OFL], 0 ; AND THEN RESET THE OVERFLOW FLAG
00006A9D 8B0D[988A0100]     <1>         mov     ecx, [TIMER_LH]         ; GET COUNT OF TIME
00006AA3 C3                 <1>       retn
                            <1>
                            <1> RTC_10:                         ; SET TIME COUNT
00006AA4 890D[988A0100]     <1>         mov     [TIMER_LH], ecx         ; SET TIME COUNT
00006AAA C605[9C8A0100]00   <1>       mov   byte [TIMER_OFL], 0 ; RESET OVERFLOW FLAG
00006AB1 C3                 <1>       retn                      ; RETURN WITH NO CARRY
                            <1>
                            <1> RTC_20:                         ; GET RTC TIME
00006AB2 E8EB010000         <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006AB7 7227               <1>       jc    short RTC_29        ; EXIT IF ERROR (CY= 1)
                            <1>
00006AB9 B000               <1>       mov   al, CMOS_SECONDS    ; SET ADDRESS OF SECONDS
00006ABB E8FD010000         <1>       call  CMOS_READ           ; GET SECONDS
```

```
00006AC0 88C6              <1>       mov   dh, al              ; SAVE
00006AC2 B00B              <1>       mov   al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
00006AC4 E8F4010000        <1>       call  CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
00006AC9 2401              <1>       and   al, 00000001b       ; MASK FOR VALID DSE BIT
00006ACB 88C2              <1>       mov   dl, al              ; SET [DL] TO ZERO FOR NO DSE BIT
00006ACD B002              <1>       mov   al, CMOS_MINUTES    ; SET ADDRESS OF MINUTES
00006ACF E8E9010000        <1>       call  CMOS_READ           ; GET MINUTES
00006AD4 88C1              <1>       mov   cl, al              ; SAVE
00006AD6 B004              <1>         mov     al, CMOS_HOURS          ; SET ADDRESS OF HOURS
00006AD8 E8E0010000        <1>       call  CMOS_READ           ; GET HOURS
00006ADD 88C5              <1>       mov   ch, al              ; SAVE
00006ADF F8                <1>       clc                       ; SET CY= 0
00006AE0                   <1> RTC_29:
00006AE0 C3                <1>       retn                      ; RETURN WITH RESULT IN CARRY FLAG
00006AE1                   <1>
00006AE1                   <1> RTC_30:                         ; SET RTC TIME
00006AE1 E8BC010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006AE6 7305              <1>       jnc   short RTC_35        ; GO AROUND IF CLOCK OPERATING
00006AE8 E817010000        <1>       call  RTC_STA                   ; ELSE TRY INITIALIZING CLOCK
00006AED                   <1> RTC_35:
00006AED 88F4              <1>       mov   ah, dh              ; GET TIME BYTE - SECONDS
00006AEF B000              <1>       mov   al, CMOS_SECONDS    ; ADDRESS SECONDS
00006AF1 E8E0010000        <1>       call  CMOS_WRITE          ; UPDATE SECONDS
00006AF6 88CC              <1>       mov   ah, cl              ; GET TIME BYTE - MINUTES
00006AF8 B002              <1>       mov   al, CMOS_MINUTES    ; ADDRESS MINUTES
00006AFA E8D7010000        <1>       call  CMOS_WRITE          ; UPDATE MINUTES
00006AFF 88EC              <1>       mov   ah, ch              ; GET TIME BYTE - HOURS
00006B01 B004              <1>       mov   al, CMOS_HOURS            ; ADDRESS HOURS
00006B03 E8CE010000        <1>       call  CMOS_WRITE          ; UPDATE ADDRESS
00006B08                   <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
00006B08                   <1>       ;mov  ah, al
00006B08 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
00006B0C E8AC010000        <1>       call  CMOS_READ           ; READ CURRENT TIME
00006B11 2462              <1>       and   al, 01100010b       ; MASK FOR VALID BIT POSITIONS
00006B13 0C02              <1>       or    al, 00000010b       ; TURN ON 24 HOUR MODE
00006B15 80E201            <1>       and   dl, 00000001b       ; USE ONLY THE DSE BIT
00006B18 08D0              <1>       or    al, dl              ; GET DAY LIGHT SAVINGS TIME BIT (OSE)
00006B1A 86E0              <1>       xchg  ah, al              ; PLACE IN WORK REGISTER AND GET ADDRESS
00006B1C E8B5010000        <1>       call  CMOS_WRITE          ; SET NEW ALARM SITS
00006B21 F8                <1>       clc                       ; SET CY= 0
00006B22 C3                <1>       retn                      ; RETURN WITH CY= 0
00006B23                   <1>
00006B23                   <1> RTC_40:                         ; GET RTC DATE
00006B23 E87A010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006B28 7225              <1>       jc    short RTC_49        ; EXIT IF ERROR (CY= 1)
00006B2A                   <1>
00006B2A B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH
00006B2C E88C010000        <1>       call  CMOS_READ           ; READ DAY OF MONTH
00006B31 88C2              <1>       mov   dl, al              ; SAVE
00006B33 B008              <1>       mov   al, CMOS_MONTH            ; ADDRESS MONTH
00006B35 E883010000        <1>       call  CMOS_READ           ; READ MONTH
00006B3A 88C6              <1>       mov   dh, al              ; SAVE
00006B3C B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR
00006B3E E87A010000        <1>       call  CMOS_READ           ; READ YEAR
00006B43 88C1              <1>       mov   cl, al              ; SAVE
00006B45 B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY LOCATION
00006B47 E871010000        <1>       call  CMOS_READ           ; GET CENTURY BYTE
00006B4C 88C5              <1>       mov   ch, al              ; SAVE
00006B4E F8                <1>       clc                       ; SET CY=0
00006B4F                   <1> RTC_49:
00006B4F C3                <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAG
00006B50                   <1>
00006B50                   <1> RTC_50:                         ; SET RTC DATE
00006B50 E84D010000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006B55 7305              <1>       jnc   short RTC_55        ; GO AROUND IF NO ERROR
00006B57 E8A8000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
00006B5C                   <1> RTC_55:
00006B5C 66B80600          <1>       mov   ax, CMOS_DAY_WEEK   ; ADDRESS OF DAY OF WEEK BYTE
00006B60 E871010000        <1>       call  CMOS_WRITE          ; LOAD ZEROS TO DAY OF WEEK
00006B65 88D4              <1>       mov   ah, dl              ; GET DAY OF MONTH BYTE
00006B67 B007              <1>       mov   al, CMOS_DAY_MONTH  ; ADDRESS DAY OF MONTH BYTE
00006B69 E868010000        <1>       call  CMOS_WRITE          ; WRITE OF DAY OF MONTH REGISTER
00006B6E 88F4              <1>       mov   ah, dh              ; GET MONTH
00006B70 B008              <1>       mov   al, CMOS_MONTH            ; ADDRESS MONTH BYTE
00006B72 E85F010000        <1>       call  CMOS_WRITE          ; WRITE MONTH REGISTER
00006B77 88CC              <1>       mov   ah, cl              ; GET YEAR BYTE
00006B79 B009              <1>       mov   al, CMOS_YEAR       ; ADDRESS YEAR REGISTER
00006B7B E856010000        <1>       call  CMOS_WRITE          ; WRITE YEAR REGISTER
00006B80 88EC              <1>       mov   ah, ch              ; GET CENTURY BYTE
00006B82 B032              <1>       mov   al, CMOS_CENTURY    ; ADDRESS CENTURY BYTE
00006B84 E84D010000        <1>       call  CMOS_WRITE          ; WRITE CENTURY LOCATION
00006B89                   <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
00006B89                   <1>       ;mov  ah, al
00006B89 66B80B0B          <1>       mov   ax, CMOS_REG_B * 257
00006B8D E82B010000        <1>       call  CMOS_READ           ; READ WIRRENT SETTINGS
00006B92 247F              <1>       and   al, 07Fh            ; CLEAR 'SET BIT'
00006B94 86E0              <1>       xchg  ah, al              ; MOVE TO WORK REGISTER
00006B96 E83B010000        <1>       call  CMOS_WRITE          ; AND START CLOCK UPDATING
00006B9B F8                <1>       clc                       ; SET CY= 0
00006B9C C3                <1>       retn                      ; RETURN CY=0
00006B9D                   <1>
00006B9D                   <1> RTC_60:                         ; SET RTC ALARM
00006B9D B00B              <1>       mov   al, CMOS_REG_B            ; ADDRESS ALARM
00006B9F E819010000        <1>       call  CMOS_READ           ; READ ALARM REGISTER
00006BA4 A820              <1>       test  al, 20h                   ; CHECK FOR ALARM ALREADY ENABLED
00006BA6 F9                <1>       stc                       ; SET CARRY IN CASE OF ERROR
00006BA7 7542              <1>       jnz   short RTC_69        ; ERROR EXIT IF ALARM SET
00006BA9 E8F4000000        <1>       call  UPD_IPR                   ; CHECK FOR UPDATE IN PROCESS
00006BAE 7305              <1>       jnc   short RTC_65        ; SKIP INITIALIZATION IF NO ERROR
00006BB0 E84F000000        <1>       call  RTC_STA                   ; ELSE INITIALIZE CLOCK
00006BB5                   <1> RTC_65:
00006BB5 88F4              <1>       mov   ah, dh              ; GET SECONDS BYTE
00006BB7 B001              <1>       mov   al, CMOS_SEC_ALARM  ; ADDRESS THE SECONDS ALARM REGISTER
00006BB9 E818010000        <1>       call  CMOS_WRITE          ; INSERT SECONDS
00006BBE 88CC              <1>       mov   ah, cl              ; GET MINUTES PARAMETER
```

```
                        <1>       mov   al, CMOS_MIN_ALARM  ; ADDRESS MINUTES ALARM REGISTER
                        <1>       call  CMOS_WRITE          ; INSERT MINUTES
                        <1>       mov   ah, ch              ; GET HOURS PARAMETER
                        <1>       mov   al, CMOS_HR_ALARM   ; ADDRESS HOUR ALARM REGISTER
                        <1>       call  CMOS_WRITE          ; INSERT HOURS
                        <1>       in    al, INTB01          ; READ SECOND INTERRUPT MASK REGISTER
                        <1>       and   al, 0FEh            ; ENABLE ALARM TIMER BIT (CY= 0)
                        <1>       out   INTB01, al          ; WRITE UPDATED MASK
                        <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
                        <1>       ;mov  ah, al
                        <1>       mov   ax, CMOS_REG_B * 257
                        <1>       call  CMOS_READ           ; READ CURRENT ALARM REGISTER
                        <1>       and   al, 07Fh            ; ENSURE SET BIT TURNED OFF
                        <1>       or    al, 20h                   ; TURN ON ALARM ENABLE
                        <1>       xchg  ah, al              ; MOVE MASK TO OUTPUT REGISTER
                        <1>       call  CMOS_WRITE          ; WRITE NEW ALARM MASK
                        <1>       clc                       ; SET CY= 0
                        <1> RTC_69:
                        <1>       mov   ax, 0               ; CLEAR AX REGISTER
                        <1>       retn                      ; RETURN WITH RESULTS IN CARRY FLAC
                        <1>
                        <1> RTC_70:                         ; RESET ALARM
                        <1>       ;mov  al, CMOS_REG_B            ; ADDRESS ALARM REGISTER
                        <1>       ;mov  ah, al
                        <1>       mov   ax, CMOS_REG_B * 257      ; ADDRESS ALARM REGISTER (TO BOTH AH,AL)
                        <1>       call  CMOS_READ           ; READ ALARM REGISTER
                        <1>       and   al, 57h                   ; TURN OFF ALARM ENABLE
                        <1>       xchg  ah, al              ; SAVE DATA AND RECOVER ADDRESS
                        <1>       call  CMOS_WRITE          ; RESTORE NEW VALUE
                        <1>       clc                       ; SET CY= 0
                        <1>       retn                      ; RETURN WITH NO CARRY
                        <1>
                        <1> RTC_STA:                ; INITIALIZE REAL TIME CLOCK
                        <1>       ;mov  al, CMOS_REG_A            ; ADDRESS REGISTER A AND LOAD DATA MASK
                        <1>       ;mov  ah, 26h
                        <1>       mov   ax, (26h*100h)+CMOS_REG_A
                        <1>       call  CMOS_WRITE          ; INITIALIZE STATUS REGISTER A
                        <1>       ;mov  al, CMOS_REG_B            ; SET "SET BIT" FOR CLOCK INITIALIZATION
                        <1>       ;mov  ah, 82h
                        <1>       mov   ax, (82h*100h)+CMOS_REG_B
                        <1>       call  CMOS_WRITE          ; AND 24 HOUR MODE TO REGISTER B
                        <1>       mov   al, CMOS_REG_C            ; ADDRESS REGISTER C
                        <1>       call  CMOS_READ           ; READ REGISTER C TO INITIALIZE
                        <1>       mov   al, CMOS_REG_D            ; ADDRESS REGISTER D
                        <1>       call  CMOS_READ           ; READ REGISTER D TO INITIALIZE
                        <1>       retn
                        <1>
                        <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>
                        <1> ;--- HARDWARE INT  70 H -- ( IRQ LEVEL  8) ------------------------------------
                        <1> ; ALARM INTERRUPT HANDLER (RTC)                                              :
                        <1> ;       THIS ROUTINE HANDLES THE PERIODIC AND ALARM INTERRUPTS FROM THE CMOS  :
                        <1> ;       TIMER. INPUT FREQUENCY IS 1.024 KHZ OR APPROXIMATELY 1024 INTERRUPTS  :
                        <1> ;       EVERY SECOND FOR THE PERIODIC INTERRUPT. FOR THE ALARM FUNCTION,      :
                        <1> ;       THE INTERRUPT WILL OCCUR AT THE DESIGNATED TIME.                      :
                        <1> ;                                                                            :
                        <1> ;       INTERRUPTS ARE ENABLED WHEN THE EVENT OR ALARM FUNCTION IS ACTIVATED. :
                        <1> ;       FOR THE EVENT INTERRUPT, THE HANDLER WILL DECREMENT THE WAIT COUNTER  :
                        <1> ;       AND WHEN IT EXPIRES WILL SET THE DESIGNATED LOCATION TO 80H. FOR      :
                        <1> ;       THE ALARM INTERRUPT. THE USER MUST PROVIDE A ROUTINE TO INTERCEPT     :
                        <1> ;       THE CORRECT ADDRESS FROM THE VECTOR TABLE INVOKED BY INTERRUPT 4AH     :
                        <1> ;       PRIOR TO SETTING THE REAL TIME CLOCK ALARM (INT 1AH, AH= 06H).        :
                        <1> ;-------------------------------------------------------------------------------
                        <1>
                        <1> RTC_A_INT: ; 07/01/2017
                        <1> ;RTC_INT:                       ; ALARM INTERRUPT
                        <1>       push  ds                  ; LEAVE INTERRUPTS DISABLED
                        <1>       push  eax                 ; SAVE REGISTERS
                        <1>       push  edi
                        <1> RTC_I_1:                        ; CHECK FOR SECOND INTERRUPT
                        <1>       mov   ax, 256*(CMOS_REG_B+NMI)+CMOS_REG_C+NMI ; ALARM AND STATUS
                        <1>       out   CMOS_PORT, al       ; WRITE ALARM FLAG MASK ADDRESS
                        <1>       nop                       ; I/O DELAY
                        <1>       jmp   short $+2
                        <1>       in    al, CMOS_DATA       ; READ AND RESET INTERRUPT REQUEST FLAGS
                        <1>       test  al, 01100000b       ; CHECK FOR EITHER INTERRUPT PENDING
                        <1>       jz    short  RTC_I_9            ; EXIT IF NOT A VALID RTC INTERRUPT
                        <1>
                        <1>       xchg  ah, al              ; SAVE FLAGS AND GET ENABLE ADDRESS
                        <1>       out   CMOS_PORT, al       ; WRITE ALARM ENABLE MASK ADDRESS
                        <1>       nop                       ; I/O DELAY
                        <1>       jmp   short $+2
                        <1>       in    al, CMOS_DATA       ; READ CURRENT ALARM ENABLE MASK
                        <1>       and   al, ah              ; ALLOW ONLY SOURCES THAT ARE ENABLED
                        <1>       test  al, 01000000b       ; CHECK FOR PERIODIC INTERRUPT
                        <1>       jz    short RTC_I_5       ; SKIP IF NOT A PERIODIC INTERRUPT
                        <1>
                        <1> ;-----      DECREMENT WAIT COUNT BY INTERRUPT INTERVAL
                        <1>
                        <1>       mov   di, KDATA           ; kernel data segment
                        <1>       mov   ds, di
                        <1>
                        <1>       sub   dword [RTC_LH], 976 ; DECREMENT COUNT BY 1/1024
                        <1>
                        <1>       jnc   short RTC_I_5       ; SKIP TILL 32 BIT WORD LESS THAN ZERO
                        <1>
                        <1> ;-----      TURN OFF PERIODIC INTERRUPT ENABLE
                        <1>
                        <1>       push  ax                  ; SAVE INTERRUPT FLAG MASK
                        <1>       mov   ax, 257*(CMOS_REG_B+NMI) ; INTERRUPT ENABLE REGISTER
                        <1>       out   CMOS_PORT, al       ; WRITE ADDRESS TO CMOS CLOCK
                        <1>       nop                       ; I/O DELAY
                        <1>       jmp   short $+2
                        <1>       in    al, CMOS_DATA       ; READ CURRENT ENABLES
                        <1>       and   al, 0BFh            ; TURN OFF PIE
```

```
00006C67 86C4               <1>       xchg   al, ah              ; GET CMOS ADDRESS AND SAVE VALUE
00006C69 E670               <1>       out    CMOS_PORT, al       ; ADDRESS REGISTER B
00006C6B 86C4               <1>       xchg   al, ah              ; GET NEW INTERRUPT ENABLE MASK
00006C6D E671               <1>       out    CMOS_DATA, al       ; SET MASK IN INTERRUPT ENABLE REGISTER
00006C6F C605[948A0100]00   <1>       mov    byte [RTC_WAIT_FLAG], 0    ; SET FUNCTION ACTIVE FLAG OFF
00006C76 8B3D[958A0100]     <1>       mov    edi, [USER_FLAG]    ; SET UP (DS:DI) TO POINT TO USER FLAG
00006C7C C60780             <1>       mov    byte [edi], 80h           ; TURN ON USERS FLAG
00006C7F 6658               <1>       pop    ax                  ; GET INTERRUPT SOURCE BACK
                            <1> RTC_I_5:
00006C81 A820               <1>       test   al, 00100000b       ; TEST FOR ALARM INTERRUPT
00006C83 740D               <1>       jz     short RTC_I_7       ; SKIP USER INTERRUPT CALL IF NOT ALARM
                            <1>
00006C85 B00D               <1>       mov    al, CMOS_REG_D            ; POINT TO DEFAULT READ ONLY REGISTER
00006C87 E670               <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
00006C89 FB                 <1>       sti                        ; INTERRUPTS BACK ON NOW
00006C8A 52                 <1>       push   edx
00006C8B E80DC10000         <1>       call   INT4Ah              ; TRANSFER TO USER ROUTINE
00006C90 5A                 <1>       pop    edx
00006C91 FA                 <1>       cli                        ; BLOCK INTERRUPT FOR RETRY
                            <1> RTC_I_7:                           ; RESTART ROUTINE TO HANDLE DELAYED
00006C92 EB94               <1>       jmp    short RTC_I_1       ;  ENTRY AND SECOND EVENT BEFORE DONE
                            <1>
                            <1> RTC_I_9:                           ; EXIT - NO PENDING INTERRUPTS
00006C94 B00D               <1>       mov    al, CMOS_REG_D            ; POINT TO DEFAULT READ ONLY REGISTER
00006C96 E670               <1>       out    CMOS_PORT, al       ; ENABLE NMI AND CMOS ADDRESS TO DEFAULT
00006C98 B020               <1>       mov    al, EOI                   ; END OF INTERRUPT MASK TO 8259 - 2
00006C9A E6A0               <1>       out    INTB00, al          ; TO 8259 - 2
00006C9C E620               <1>       out    INTA00,      al           ; TO 8259 - 1
00006C9E 5F                 <1>       pop    edi                 ; RESTORE REGISTERS
00006C9F 58                 <1>       pop    eax
00006CA0 1F                 <1>       pop    ds
00006CA1 CF                 <1>       iretd                      ; END OF INTERRUPT
                            <1>
                            <1>
                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (bios2.asm)
                            <1> UPD_IPR:                           ; WAIT TILL UPDATE NOT IN PROGRESS
00006CA2 51                 <1>       push   ecx
                            <1>
                            <1>       ; 29/05/2016
00006CA3 B968110000         <1>       mov    ecx, ((1984+244)*4)/2     ; AWARD BIOS 1999, ATIME.ASM
                            <1>                                  ; 'WAITCPU_CK_UD_STAT'
                            <1>                                  ; (244Us + 1984Us)
                            <1>                                  ; (assume each read takes
                            <1>                                  ;  2 microseconds).
                            <1>       ;mov   ecx, 65535
                            <1>             ;mov cx, 800         ; SET TIMEOUT LOOP COUNT (= 800)
                            <1> UPD_10:
00006CA8 B00A               <1>       mov    al, CMOS_REG_A            ; ADDRESS STATUS REGISTER A
00006CAA FA                 <1>       cli                        ; NO TIMER INTERRUPTS DURING UPDATES
00006CAB E80D000000         <1>       call   CMOS_READ           ; READ UPDATE IN PROCESS FLAG
00006CB0 A880               <1>       test   al, 80h                   ; IF UIP BIT IS ON ( CANNOT READ TIME )
00006CB2 7406               <1>       jz     short UPD_90        ; EXIT WITH CY= 0 IF CAN READ CLOCK NOW
00006CB4 FB                 <1>       sti                        ; ALLOW INTERRUPTS WHILE WAITING
00006CB5 E2F1               <1>       loop   UPD_10              ; LOOP TILL READY OR TIMEOUT
00006CB7 31C0               <1>       xor    eax, eax            ; CLEAR RESULTS IF ERROR
                            <1>             ; xor ax, ax
00006CB9 F9                 <1>       stc                        ; SET CARRY FOR ERROR
                            <1> UPD_90:
00006CBA 59                 <1>       pop    ecx                 ; RESTORE CALLERS REGISTER
00006CBB FA                 <1>       cli                        ; INTERRUPTS OFF DURING SET
00006CBC C3                 <1>       retn                       ; RETURN WITH CY FLAG SET
                            <1>
                            <1>
                            <1>       ; 29/05/2016 - TRDOS 386 (TRDOS v2.0)
                            <1>       ; 22/08/2014 (Retro UNIX 386 v1)
                            <1>       ; IBM PC/AT BIOS source code ----- 10/06/85 (test4.asm)
                            <1>
                            <1> ;--- CMOS_READ --------------------------------------------------------------------
                            <1> ;           READ BYTE FROM CMOS_SYSTEM CLOCK CONFIGURATION TABLE       :
                            <1> ;                                                                      :
                            <1> ; INPUT: (AL)=     CMOS_TABLE ADDRESS TO BE READ                       :
                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO READ               :
                            <1> ;                                                                      :
                            <1> ; OUTPUT: (AL)     VALUE AT LOCATION (AL) MOVED INTO (AL). IF BIT 7 OF (AL) WAS   :
                            <1> ;           ON THEN NMI LEFT DISABLED, DURING THE CMOS READ BOTH NMI AND   :
                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.      :
                            <1> ;           ONLY THE (AL) REGISTER AND THE NMI STATE IS CHANGED.       :
                            <1> ;-------------------------------------------------------------------------------
                            <1>
                            <1> CMOS_READ:
00006CBD 9C                 <1>       pushf                      ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
00006CBE D0C0               <1>       rol    al, 1               ; MOVE NMI BIT TO LOW POSITION
00006CC0 F9                 <1>       stc                        ; FORCE NMI BIT ON IN CARRY FLAG
00006CC1 D0D8               <1>       rcr    al, 1               ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
00006CC3 FA                 <1>       cli                        ; DISABLE INTERRUPTS
00006CC4 E670               <1>       out    CMOS_PORT, al       ; ADDRESS LOCATION AND DISABLE NMI
                            <1>       ; 29/05/2016
                            <1>       ;nop                       ; I/O DELAY
00006CC6 E6EB               <1>       out    0ebh,al      ; NEWIODELAY ; AWARD BIOS 1999, ATIME.ASM
                            <1>       ;
00006CC8 E471               <1>       in     al, CMOS_DATA       ; READ THE REQUESTED CMOS LOCATION
00006CCA 6650               <1>       push   ax                  ; SAVE (AH) REGISTER VALUE AND CMOS BYTE
                            <1>       ; 15/03/2015 ; IBM PC/XT Model 286 BIOS source code
                            <1>                    ; ----- 10/06/85 (test4.asm)
00006CCC B01E               <1>       mov    al, CMOS_SHUT_DOWN*2       ; GET ADDRESS OF DEFAULT LOCATION
                            <1>       ;mov   al, CMOS_REG_D*2    ; GET ADDRESS OF DEFAULT LOCATION
00006CCE D0D8               <1>       rcr    al, 1               ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00006CD0 E670               <1>       out    CMOS_PORT, al       ; SET DEFAULT TO READ ONLY REGISTER
00006CD2 6658               <1>       pop    ax                  ; RESTORE (AH) AND (AL), CMOS BYTE
00006CD4 9D                 <1>       popf
```

```
00006CD5 C3                 <1>       retn                        ; RETURN WITH FLAGS RESTORED
                            <1>
                            <1> ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>
                            <1> ;--- CMOS_WRITE --------------------------------------------------------------------
                            <1> ;     WRITE BYTE TO CMOS SYSTEM CLOCK CONFIGURATION TABLE               :
                            <1> ;                                                                       :
                            <1> ; INPUT: (AL)=     CMOS TABLE ADDRESS TO BE WRITTEN TO                          :
                            <1> ;           BIT    7 = 0 FOR NMI ENABLED AND 1 FOR NMI DISABLED ON EXIT    :
                            <1> ;           BITS 6-0 = ADDRESS OF TABLE LOCATION TO WRITE               :
                            <1> ;      (AH)= NEW VALUE TO BE PLACED IN THE ADDRESSED TABLE LOCATION            :
                            <1> ;                                                                       :
                            <1> ; OUTPUT:   VALUE IN (AH) PLACED IN LOCATION (AL) WITH NMI LEFT DISABLED   :
                            <1> ;           IF BIT 7 OF (AL) IS ON, DURING THE CMOS UPDATE BOTH NMI AND    :
                            <1> ;           NORMAL INTERRUPTS ARE DISABLED TO PROTECT CMOS DATA INTEGRITY. :
                            <1> ;           THE CMOS ADDRESS REGISTER IS POINTED TO A DEFAULT VALUE AND    :
                            <1> ;           THE INTERRUPT FLAG RESTORED TO THE ENTRY STATE ON RETURN.      :
                            <1> ;           ONLY THE CMOS LOCATION AND THE NMI STATE IS CHANGED.       :
                            <1> ;-----------------------------------------------------------------------------
                            <1>
                            <1> CMOS_WRITE:                     ; WRITE (AH) TO LOCATION (AL)
00006CD6 9C                 <1>       pushf                     ; SAVE INTERRUPT ENABLE STATUS AND FLAGS
00006CD7 6650               <1>       push   ax                 ; SAVE WORK REGISTER VALUES
00006CD9 D0C0               <1>       rol    al, 1              ; MOVE NMI BIT TO LOW POSITION
00006CDB F9                 <1>       stc                       ; FORCE NMI BIT ON IN CARRY FLAG
00006CDC D0D8               <1>       rcr    al, 1              ; HIGH BIT ON TO DISABLE NMI - OLD IN CY
00006CDE FA                 <1>       cli                       ; DISABLE INTERRUPTS
00006CDF E670               <1>       out    CMOS_PORT, al      ; ADDRESS LOCATION AND DISABLE NMI
00006CE1 88E0               <1>       mov    al, ah             ; GET THE DATA BYTE TO WRITE
00006CE3 E671               <1>       out    CMOS_DATA, al      ; PLACE IN REQUESTED CMOS LOCATION
00006CE5 B01E               <1>       mov    al, CMOS_SHUT_DOWN*2      ; GET ADDRESS OF DEFAULT LOCATION
                            <1>       ;mov   al, CMOS_REG_D*2   ; GET ADDRESS OF DEFAULT LOCATION
00006CE7 D0D8               <1>       rcr    al, 1              ; PUT ORIGINAL NMI MASK BIT INTO ADDRESS
00006CE9 E670               <1>       out    CMOS_PORT, al      ; SET DEFAULT TO READ ONLY REGISTER
00006CEB 90                 <1>       nop                       ; I/O DELAY
00006CEC E471               <1>       in     al, CMOS_DATA      ; OPEN STANDBY LATCH
00006CEE 6658               <1>       pop    ax                 ; RESTORE WORK REGISTERS
00006CF0 9D                 <1>       popf
00006CF1 C3                 <1>       retn
                            <1>
                            <1> ; /// End Of TIMER FUNCTIONS ///

00006CF2 90<rept>               Align 16

                                gdt:  ; Global Descriptor Table
                                      ; 02/12/2020
                                      ; (30/07/2015, conforming cs)
                                      ; (26/03/2015)
                                      ; (24/03/2015, tss)
                                      ; (19/03/2015)
                                      ; (29/12/2013)
                                      ;
00006D00 0000000000000000             dw 0, 0, 0, 0       ; NULL descriptor
                                gdt_kcode:
                                      ; 18/08/2014
                                                ; 8h kernel code segment, base = 00000000h
                                      ;dw 0FFFFh, 0, 9E00h, 00CFh      ; KCODE  ; 30/12/2016
00006D08 FFFF0000009ACF00             dw 0FFFFh, 0, 9A00h, 00CFh; KCODE
                                gdt_kdata:
                                                ; 10h kernel data segment, base = 00000000h
00006D10 FFFF00000092CF00             dw 0FFFFh, 0, 9200h, 00CFh; KDATA
                                gdt_ucode:
                                                ; 1Bh user code segment, base address = 400000h ; CORE
                                      ;dw 0FBFFh, 0, 0FE40h, 00CFh     ; UCODE  ; 30/12/2016
00006D18 FFFB000040FACF00             dw 0FBFFh, 0, 0FA40h, 00CFh      ; UCODE
                                gdt_udata:
                                                ; 23h user data segment, base address = 400000h ; CORE
00006D20 FFFB000040F2CF00             dw 0FBFFh, 0, 0F240h, 00CFh      ; UDATA
                                gdt_tss:
                                                ; Task State Segment
00006D28 6700                         dw 0067h ; Limit = 103 ; (104-1, tss size = 104 byte,
                                                               ;  no IO permission in ring 3)
                                gdt_tss0:
00006D2A 0000                         dw 0  ; TSS base address, bits 0-15
                                gdt_tss1:
00006D2C 00                           db 0  ; TSS base address, bits 16-23
                                                ; 49h
00006D2D E9                           db 11101001b ; 0E9h => P=1/DPL=11/0/1/0/B/1 --> B = Task is busy (1)
00006D2E 00                           db 0 ; G/0/0/AVL/LIMIT=0000 ; (Limit bits 16-19 = 0000) (G=0, 1 byte)
                                gdt_tss2:
00006D2F 00                           db 0  ; TSS base address, bits 24-31

                                      ; 30/11/2020
                                      ; 29/11/2020 - TRDOS v2.0.3
                                      ; VESA VBE3 VIDE BIOS 32 BIT PMI SEGMENTS (16 bit segments)
                                                ; 30h ; VBE3CS
                                _vbe3_CS:  ; vesa vbe3 bios uses this as code seg (same addr with _vbe3_DS)
                                      ; limit = 65536, base addr = 0, P/DPL/1/Type/C/R/A = 9Ah, 16 bit
00006D30 FFFF0000009A0000             dw 0FFFFh, 0, 9A00h, 0 ; Note: base addr will be initialized
                                                ; 38h ; VBE3BDS
                                _vbe3_BDS: ; vesa vbe3 bios uses this as equivalent of rombios data segment
                                      ; limit = 1536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006D38 FF05000000920000             dw 05FFh, 0, 9200h, 0 ; Note: base addr will be initialized
                                                ; 40h ; VBE3A000
                                _A0000Sel: ; VGA default video memory address
                                      ; limit = 65536, base addr = 0A0000h, 16 bit
00006D40 FFFF00000A920000             dw 0FFFFh, 0, 920Ah, 0
                                                ; 48h ; VBE3B000
                                _B0000Sel: ; MDA (monochrome) video memory address
                                      ; limit = 65536, base addr = 0B0000h, 16 bit
00006D48 FFFF00000B920000             dw 0FFFFh, 0, 920Bh, 0
                                                ; 50h ; VBE3B800
                                _B8000Sel: ; CGA video memory address
                                      ; limit = 32768, base addr = 0B8000h, 16 bit
00006D50 FF7F00800B920000             dw 07FFFh, 8000h, 920Bh, 0
```

```
                                        ; 58h ; VBE3DS
                                _vbe3_DS: ; vesa vbe3 bios uses this as data seg (CodeSegSel in PMInfoBlock)
                                        ; limit = 65536, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006D58 FFFF000000920000               dw 0FFFFh, 0, 9200h, 0 ; Note: base addr will be initialized
                                        ; 60h ; VBE3SS
                                _vbe3_SS: ; kernel's stack segment but 16 bit version (same stack addr)
                                        ; limit = 1024, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006D60 FF03000000920000               dw 03FFh, 0, 9200h, 0 ; Note: base addr will be initialized
                                        ; 68h ; VBE3ES
                                _vbe3_ES: ; extra 16 bit segment points to buffers in kernel's mem space
                                        ; limit = 2048, base addr = 0, P/DPL/1/Type/E/W/A = 92h, 16 bit
00006D68 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
00006D70 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:
00006D78 7700                           dw gdt_end - gdt - 1    ; Limit (size)
00006D7A [006D0000]                     dd gdt                  ; Address of the GDT

                                        ; 20/08/2014
                                idtd:
00006D7E 7F02                           dw idt_end - idt - 1    ; Limit (size)
00006D80 [30870100]                     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
                              <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>
00006D84 [E76D0000]           <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:
00006D88 01             <1>             DB     01           ;DRIVE TYPE, MEDIA TABLE
                        <1>                ;DW      MD_TBL1
00006D89 [A66D0000]     <1>             dd     MD_TBL1
00006D8D 82             <1>             DB     02+BIT7ON
                        <1>             ;DW      MD_TBL2
00006D8E [B36D0000]     <1>                dd      MD_TBL2
00006D92 02             <1> DR_DEFAULT:  DB     02
                        <1>                ;DW      MD_TBL3
00006D93 [C06D0000]     <1>             dd      MD_TBL3
00006D97 03             <1>             DB     03
                        <1>                ;DW      MD_TBL4
00006D98 [CD6D0000]     <1>             dd      MD_TBL4
00006D9C 84             <1>             DB     04+BIT7ON
                        <1>                ;DW      MD_TBL5
00006D9D [DA6D0000]     <1>             dd      MD_TBL5
00006DA1 04             <1>             DB     04
                        <1>                ;DW      MD_TBL6
00006DA2 [E76D0000]     <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:
00006DA6 DF             <1>       DB     11011111B    ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
```

```
00006DA7 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DA8 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DA9 02                <1>       DB     2             ; 512 BYTES/SECTOR
00006DAA 09                <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006DAB 2A                <1>       DB     02AH          ; GAP LENGTH
00006DAC FF                <1>       DB     0FFH          ; DTL
00006DAD 50                <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006DAE F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DAF 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DB0 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DB1 27                <1>       DB     39            ; MAX. TRACK NUMBER
00006DB2 80                <1>       DB     RATE_250      ; DATA TRANSFER RATE
                           <1> ;--------------------------------------------------------
                           <1> ;     360 KB MEDIA IN 1.2 MB DRIVE                   :
                           <1> ;--------------------------------------------------------
                           <1> MD_TBL2:
00006DB3 DF                <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DB4 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DB5 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DB6 02                <1>       DB     2             ; 512 BYTES/SECTOR
00006DB7 09                <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006DB8 2A                <1>       DB     02AH          ; GAP LENGTH
00006DB9 FF                <1>       DB     0FFH          ; DTL
00006DBA 50                <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006DBB F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DBC 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DBD 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DBE 27                <1>       DB     39            ; MAX. TRACK NUMBER
00006DBF 40                <1>       DB     RATE_300      ; DATA TRANSFER RATE
                           <1> ;--------------------------------------------------------
                           <1> ;     1.2 MB MEDIA IN 1.2 MB DRIVE                   :
                           <1> ;--------------------------------------------------------
                           <1> MD_TBL3:
00006DC0 DF                <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DC1 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DC2 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DC3 02                <1>       DB     2             ; 512 BYTES/SECTOR
00006DC4 0F                <1>       DB     15            ; EOT (LAST SECTOR ON TRACK)
00006DC5 1B                <1>       DB     01BH          ; GAP LENGTH
00006DC6 FF                <1>       DB     0FFH          ; DTL
00006DC7 54                <1>       DB     054H          ; GAP LENGTH FOR FORMAT
00006DC8 F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DC9 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DCA 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DCB 4F                <1>       DB     79            ; MAX. TRACK NUMBER
00006DCC 00                <1>       DB     RATE_500      ; DATA TRANSFER RATE
                           <1> ;--------------------------------------------------------
                           <1> ;     720 KB MEDIA IN 720 KB DRIVE                   :
                           <1> ;--------------------------------------------------------
                           <1> MD_TBL4:
00006DCD DF                <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DCE 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DCF 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DD0 02                <1>       DB     2             ; 512 BYTES/SECTOR
00006DD1 09                <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006DD2 2A                <1>       DB     02AH          ; GAP LENGTH
00006DD3 FF                <1>       DB     0FFH          ; DTL
00006DD4 50                <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006DD5 F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DD6 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DD7 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DD8 4F                <1>       DB     79            ; MAX. TRACK NUMBER
00006DD9 80                <1>       DB     RATE_250      ; DATA TRANSFER RATE
                           <1> ;--------------------------------------------------------
                           <1> ;     720 KB MEDIA IN 1.44 MB DRIVE                  :
                           <1> ;--------------------------------------------------------
                           <1> MD_TBL5:
00006DDA DF                <1>       DB     11011111B     ; SRT=D, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DDB 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DDC 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DDD 02                <1>       DB     2             ; 512 BYTES/SECTOR
00006DDE 09                <1>       DB     09            ; EOT (LAST SECTOR ON TRACK)
00006DDF 2A                <1>       DB     02AH          ; GAP LENGTH
00006DE0 FF                <1>       DB     0FFH          ; DTL
00006DE1 50                <1>       DB     050H          ; GAP LENGTH FOR FORMAT
00006DE2 F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DE3 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DE4 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DE5 4F                <1>       DB     79            ; MAX. TRACK NUMBER
00006DE6 80                <1>       DB     RATE_250      ; DATA TRANSFER RATE
                           <1> ;--------------------------------------------------------
                           <1> ;     1.44 MB MEDIA IN 1.44 MB DRIVE                 :
                           <1> ;--------------------------------------------------------
                           <1> MD_TBL6:
00006DE7 AF                <1>       DB     10101111B     ; SRT=A, HD UNLOAD=0F - 1ST SPECIFY BYTE
00006DE8 02                <1>       DB     2             ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
00006DE9 25                <1>       DB     MOTOR_WAIT    ; WAIT TIME AFTER OPERATION TILL MOTOR OFF
00006DEA 02                <1>       DB     2             ; 512 BYTES/SECTOR
00006DEB 12                <1>       DB     18            ; EOT (LAST SECTOR ON TRACK)
00006DEC 1B                <1>       DB     01BH          ; GAP LENGTH
00006DED FF                <1>       DB     0FFH          ; DTL
00006DEE 6C                <1>       DB     06CH          ; GAP LENGTH FOR FORMAT
00006DEF F6                <1>       DB     0F6H          ; FILL BYTE FOR FORMAT
00006DF0 0F                <1>       DB     15            ; HEAD SETTLE TIME (MILLISECONDS)
00006DF1 08                <1>       DB     8             ; MOTOR START TIME (1/8 SECONDS)
00006DF2 4F                <1>       DB     79            ; MAX. TRACK NUMBER
00006DF3 00                <1>       DB     RATE_500      ; DATA TRANSFER RATE
                           <1>
                           <1>
                           <1> ; << diskette.inc >>
                           <1> ; ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                           <1> ;
                           <1> ;----------------------------------------
                           <1> ;     ROM BIOS DATA AREAS       :
                           <1> ;----------------------------------------
```

```
                        <1>
                        <1> ;DATA       SEGMENT AT 40H            ; ADDRESS= 0040:0000
                        <1>
                        <1> ;----------------------------------------
                        <1> ;     FIXED DISK DATA AREAS            :
                        <1> ;----------------------------------------
                        <1>
                        <1> ;DISK_STATUS1:     DB     0            ; FIXED DISK STATUS
                        <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:
 279 00006DF4 E0        <1>       db     NO_ERR
 280 00006DF5 024001BB  <1>       db     BAD_ADDR_MARK,BAD_SEEK,BAD_CMD,UNDEF_ERR
 281 00006DF9 04BB100A  <1>       db     RECORD_NOT_FND,UNDEF_ERR,BAD_ECC,BAD_SECTOR
 282                    <1>
 283                    <1> ; 17/12/2014 (mov ax, [cfd])
 284                    <1> ; 11/12/2014
 285 00006DFD 00        <1> cfd:         db 0               ; current floppy drive (for GET_PARM)
 286                    <1> ; 17/12/2014                    ; instead of 'DISK_POINTER'
 287 00006DFE 01        <1> pfd:         db 1               ; previous floppy drive (for GET_PARM)
 288                    <1>                                 ; (initial value of 'pfd
 289                    <1>                                 ; must be different then 'cfd' value
 290                    <1>                                 ; to force updating/initializing
 291                    <1>                                 ; current drive parameters)
 292 00006DFF 90        <1> align 2
 293                    <1>
 294 00006E00 F001      <1> HF_PORT:     dw     1F0h  ; Default = 1F0h
 295                    <1>                           ; (170h)
 296 00006E02 F603      <1> HF_REG_PORT: dw     3F6h  ; HF_PORT + 206h
 297                    <1>
 298                    <1> ; 05/01/2015
 299 00006E04 00        <1> hf_m_s:         db      0     ; (0 = Master, 1 = Slave)
 300                    <1>
 301                    <1> ; ****************************************************************************
3031
3032 00006E05 90            Align 2
3033
3034                        ; 04/11/2014 (Retro UNIX 386 v1)
3035 00006E06 0000          mem_1m_1k:   dw 0  ; Number of contiguous KB between
3036                                           ; 1 and 16 MB, max. 3C00h = 15 MB.
3037 00006E08 0000          mem_16m_64k: dw 0  ; Number of contiguous 64 KB blocks
3038                                         ; between 16 MB and 4 GB.
3039
3040                        ; 12/11/2014 (Retro UNIX 386 v1)
3041 00006E0A 00            boot_drv:    db 0 ; boot drive number (physical)
3042                        ; 24/11/2014
3043 00006E0B 00            drv:       db 0
3044 00006E0C 00            last_drv:    db 0 ; last hdd
3045 00006E0D 00            hdc:         db 0  ; number of hard disk drives
3046                                          ; (present/detected)
3047
3048                        ; 24/11/2014 (Retro UNIX 386 v1)
3049                        ; Physical drive type & flags
3050 00006E0E 00            fd0_type:    db 0  ; floppy drive type
3051 00006E0F 00            fd1_type:    db 0  ; 4 = 1.44 Mb, 80 track, 3.5" (18 spt)
3052                                           ; 6 = 2.88 Mb, 80 track, 3.5" (36 spt)
3053                                           ; 3 = 720 Kb, 80 track, 3.5" (9 spt)
3054                                           ; 2 = 1.2 Mb, 80 track, 5.25" (15 spt)
3055                                           ; 1 = 360 Kb, 40 track, 5.25" (9 spt)
3056 00006E10 00            hd0_type:    db 0  ; EDD status for hd0 (bit 7 = present flag)
3057 00006E11 00            hd1_type:    db 0  ; EDD status for hd1 (bit 7 = present flag)
3058 00006E12 00            hd2_type:    db 0  ; EDD status for hd2 (bit 7 = present flag)
3059 00006E13 00            hd3_type:    db 0  ; EDD status for hd3 (bit 7 = present flag)
3060                                           ; bit 0 - Fixed disk access subset supported
3061                                           ; bit 1 - Drive locking and ejecting
3062                                           ; bit 2 - Enhanced disk drive support
3063                                           ; bit 3 = Reserved (64 bit EDD support)
3064                                           ; (If bit 0 is '1' Retro UNIX 386 v1
3065                                           ; will interpret it as 'LBA ready'!)
3066
3067                        ; 11/03/2015 - 10/07/2015
3068 00006E14 00000000000000000-   drv.cylinders: dw 0,0,0,0,0,0,0
3068 00006E1D 0000000000
3069 00006E22 00000000000000000-   drv.heads:     dw 0,0,0,0,0,0,0
3069 00006E2B 0000000000
3070 00006E30 00000000000000000-   drv.spt:       dw 0,0,0,0,0,0,0
3070 00006E39 0000000000
3071 00006E3E 00000000000000000-   drv.size:      dd 0,0,0,0,0,0,0
3071 00006E47 00000000000000000-
3071 00006E50 00000000000000000-
3071 00006E59 00
3072 00006E5A 00000000000000       drv.status:    db 0,0,0,0,0,0,0
3073 00006E61 00000000000000       drv.error:     db 0,0,0,0,0,0,0
```

```
3074
3075                                    Align 2
3076
3077                                    ;;; 11/03/2015
3078                                    %include 'kybdata.s'      ; KEYBOARD (BIOS) DATA
   1                                <1> ; ****************************************************************************
   2                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - kybdata.s
   3                                <1> ; ----------------------------------------------------------------------------
   4                                <1> ; Last Update: 17/01/2016
   5                                <1> ; ----------------------------------------------------------------------------
   6                                <1> ; Beginning: 17/01/2016
   7                                <1> ; ----------------------------------------------------------------------------
   8                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                                <1> ; ----------------------------------------------------------------------------
  10                                <1> ; Turkish Rational DOS
  11                                <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                                <1> ;
  13                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  14                                <1> ; kybdata.inc (11/03/2015)
  15                                <1> ;
  16                                <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
  17                                <1> ; ****************************************************************************
  18                                <1>
  19                                <1> ; Retro UNIX 386 v1 Kernel - KYBDATA.INC
  20                                <1> ; Last Modification: 11/03/2015
  21                                <1> ;             (Data Section for 'KEYBOARD.INC')
  22                                <1> ;
  23                                <1> ; ///////// KEYBOARD DATA ///////////////
  24                                <1>
  25                                <1> ; 05/12/2014
  26                                <1> ; 04/12/2014 (derived from pc-xt-286 bios source code -1986-)
  27                                <1> ; 03/06/86  KEYBOARD BIOS
  28                                <1>
  29                                <1> ;--------------------------------------------------------------------------------
  30                                <1> ;     KEY IDENTIFICATION SCAN TABLES
  31                                <1> ;--------------------------------------------------------------------------------
  32                                <1>
  33                                <1> ;-----       TABLES FOR ALT CASE ------------
  34                                <1> ;-----       ALT-INPUT-TABLE
  35 00006E68 524F50514B            <1> K30:  db     82,79,80,81,75
  36 00006E6D 4C4D474849            <1>       db     76,77,71,72,73            ; 10 NUMBER ON KEYPAD
  37                                <1> ;-----       SUPER-SHIFT-TABLE
  38 00006E72 101112131415          <1>       db     16,17,18,19,20,21   ; A-Z TYPEWRITER CHARS
  39 00006E78 161718191E1F          <1>       db     22,23,24,25,30,31
  40 00006E7E 202122232425          <1>       db     32,33,34,35,36,37
  41 00006E84 262C2D2E2F30          <1>       db     38,44,45,46,47,48
  42 00006E8A 3132                  <1>       db     49,50
  43                                <1>
  44                                <1> ;-----       TABLE OF SHIFT KEYS AND MASK VALUES
  45                                <1> ;-----       KEY_TABLE
  46 00006E8C 52                    <1> _K6:    db      INS_KEY                ; INSERT KEY
  47 00006E8D 3A4546381D            <1>       db     CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
  48 00006E92 2A36                  <1>         db      LEFT_KEY,RIGHT_KEY
  49                                <1> _K6L    equ     $-_K6
  50                                <1>
  51                                <1> ;-----       MASK_TABLE
  52 00006E94 80                    <1> _K7:    db      INS_SHIFT              ; INSERT MODE SHIFT
  53 00006E95 4020100804            <1>       db     CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
  54 00006E9A 0201                  <1>       db     LEFT_SHIFT,RIGHT_SHIFT
  55                                <1>
  56                                <1> ;-----       TABLES FOR CTRL CASE            ;---- CHARACTERS ------
  57 00006E9C 1BFF00FFFFFF          <1> _K8:  db     27,-1,0,-1,-1,-1    ; Esc, 1, 2, 3, 4, 5
  58 00006EA2 1EFFFFFFFF1F          <1>       db     30,-1,-1,-1,-1,31   ; 6, 7, 8, 9, 0, -
  59 00006EA8 FF7FFF111705          <1>       db     -1,127,-1,17,23,5   ; =, Bksp, Tab, Q, W, E
  60 00006EAE 12141915090F          <1>       db     18,20,25,21,9,15    ; R, T, Y, U, I, O
  61 00006EB4 101B1D0AFF01          <1>       db     16,27,29,10,-1,1    ; P, [, ], Enter, Ctrl, A
  62 00006EBA 13040607080A          <1>       db     19,4,6,7,8,10       ; S, D, F, G, H, J
  63 00006EC0 0B0CFFFFFFFF          <1>       db     11,12,-1,-1,-1,-1   ; K, L, :, ', `, LShift
  64 00006EC6 1C1A18031602          <1>       db     28,26,24,3,22,2           ; Bkslash, Z, X, C, V, B
  65 00006ECC 0E0DFFFFFFFF          <1>       db     14,13,-1,-1,-1,-1   ; N, M, ,, ., /, RShift
  66 00006ED2 96FF20FF              <1>       db     150,-1,' ',-1       ; *, ALT, Spc, CL
  67                                <1>       ;                         ;----- FUNCTIONS ------
  68 00006ED6 5E5F60616263          <1>       db     94,95,96,97,98,99   ; F1 - F6
  69 00006EDC 64656667FFFF          <1>       db     100,101,102,103,-1,-1     ; F7 - F10, NL, SL
  70 00006EE2 778D848E738F          <1>       db     119,141,132,142,115,143   ; Home, Up, PgUp, -, Left, Pad5
  71 00006EE8 749075917692          <1>       db     116,144,117,145,118,146 ; Right, +, End, Down, PgDn, Ins
  72 00006EEE 93FFFFFF898A          <1>       db     147,-1,-1,-1,137,138      ; Del, SysReq, Undef, WT, F11, F12
  73                                <1>
  74                                <1> ;-----       TABLES FOR LOWER CASE ----------
  75 00006EF4 1B3132333435363738-   <1> K10:  db     27,'1234567890-=',8,9
  75 00006EFD 39302D3D0809          <1>
  76 00006F03 71776572747975696F-   <1>       db     'qwertyuiop[]',13,-1,'asdfghjkl;',39
  76 00006F0C 705B5D0DFF61736466-   <1>
  76 00006F15 67686A6B6C3B27        <1>
  77 00006F1C 60FF5C7A786376626E-   <1>       db     96,-1,92,'zxcvbnm,./',-1,'*',-1,' ',-1
  77 00006F25 6D2C2E2FFF2AFF20FF    <1>
  78                                <1> ;-----       LC TABLE SCAN
  79 00006F2E 3B3C3D3E3F            <1>       db     59,60,61,62,63            ; BASE STATE OF F1 - F10
  80 00006F33 4041424344            <1>       db     64,65,66,67,68
  81 00006F38 FFFF                  <1>       db     -1,-1               ; NL, SL
  82                                <1>
  83                                <1> ;-----       KEYPAD TABLE
  84 00006F3A 474849FF4BFF          <1> K15:  db     71,72,73,-1,75,-1   ; BASE STATE OF KEYPAD KEYS
  85 00006F40 4DFF4F50515253        <1>       db     77,-1,79,80,81,82,83
  86 00006F47 FFFF5C8586            <1>       db     -1,-1,92,133,134    ; SysRq, Undef, WT, F11, F12
  87                                <1>
  88                                <1> ;-----       TABLES FOR UPPER CASE ----------
  89 00006F4C 1B21402324255E262A-   <1> K11:  db     27,'!@#$%',94,'&*()_+',8,0
  89 00006F55 28295F2B0800          <1>
  90 00006F5B 5157455255955494F-    <1>       db     'QWERTYUIOP{}',13,-1,'ASDFGHJKL:"'
  90 00006F64 507B7D0DFF41534446-   <1>
  90 00006F6D 47484A4B4C3A22        <1>
  91 00006F74 7EFF7C5A584356424E-   <1>       db     126,-1,'|ZXCVBNM<>?',-1,'*',-1,' ',-1
  91 00006F7D 4D3C3E3FFF2AFF20FF    <1>
  92                                <1> ;-----       UC TABLE SCAN
```

```
 93 00006F86 5455565758          <1> K12:  db     84,85,86,87,88            ; SHIFTED STATE OF F1 - F10
 94 00006F8B 595A5B5C5D          <1>       db     89,90,91,92,93
 95 00006F90 FFFF                <1>       db     -1,-1               ; NL, SL
 96                              <1>
 97                              <1> ;-----       NUM STATE TABLE
 98 00006F92 3738392D3435362B31- <1> K14:  db     '789-456+1230.'           ; NUMLOCK STATE OF KEYPAD KEYS
 98 00006F9B 3233302E            <1>
 99                              <1>       ;
100 00006F9F FFFF7C8788          <1>       db     -1,-1,124,135,136   ; SysRq, Undef, WT, F11, F12
101                              <1>
102                              <1> ; 26/08/2014
103                              <1> ; Retro UNIX 8086 v1 - UNIX.ASM (03/03/2014)
104                              <1> ; Derived from IBM "pc-at"
105                              <1> ; rombios source code (06/10/1985)
106                              <1> ; 'dseg.inc'
107                              <1>
108                              <1> ;----------------------------------------;
109                              <1> ;     SYSTEM DATA AREA          ;
110                              <1> ;----------------------------------------
111 00006FA4 00                  <1> BIOS_BREAK   db     0            ; BIT 7=1 IF BREAK KEY HAS BEEN PRESSED
112                              <1>
113                              <1> ;----------------------------------------
114                              <1> ;     KEYBOARD DATA AREAS       ;
115                              <1> ;----------------------------------------
116                              <1>
117 00006FA5 00                  <1> KB_FLAG            db     0            ; KEYBOARD SHIFT STATE AND STATUS FLAGS
118 00006FA6 00                  <1> KB_FLAG_1    db     0            ; SECOND BYTE OF KEYBOARD STATUS
119 00006FA7 00                  <1> KB_FLAG_2    db     0            ; KEYBOARD LED FLAGS
120 00006FA8 00                  <1> KB_FLAG_3    db     0            ; KEYBOARD MODE STATE AND TYPE FLAGS
121 00006FA9 00                  <1> ALT_INPUT    db     0            ; STORAGE FOR ALTERNATE KEY PAD ENTRY
122 00006FAA [BA6F0000]          <1> BUFFER_START dd     KB_BUFFER    ; OFFSET OF KEYBOARD BUFFER START
123 00006FAE [DA6F0000]          <1> BUFFER_END   dd     KB_BUFFER + 32      ; OFFSET OF END OF BUFFER
124 00006FB2 [BA6F0000]          <1> BUFFER_HEAD  dd     KB_BUFFER    ; POINTER TO HEAD OF KEYBOARD BUFFER
125 00006FB6 [BA6F0000]          <1> BUFFER_TAIL  dd     KB_BUFFER    ; POINTER TO TAIL OF KEYBOARD BUFFER
126                              <1> ; ------     HEAD = TAIL  INDICATES THAT THE BUFFER IS EMPTY
127 00006FBA 0000<rept>          <1> KB_BUFFER    times  16 dw 0             ; ROOM FOR 16 SCAN CODE ENTRIES
128                              <1>
129                              <1> ; /// End Of KEYBOARD DATA ///
3079                                 %include 'vidata.s'       ; VIDEO (BIOS) DATA
  1                              <1> ; ****************************************************************************
  2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - vidata.s
  3                              <1> ; ----------------------------------------------------------------------------
  4                              <1> ; Last Update: 24/11/2020
  5                              <1> ; ----------------------------------------------------------------------------
  6                              <1> ; Beginning: 16/01/2016
  7                              <1> ; ----------------------------------------------------------------------------
  8                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
  9                              <1> ; ----------------------------------------------------------------------------
 10                              <1> ; Turkish Rational DOS
 11                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
 12                              <1> ;
 13                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
 14                              <1> ; vidata.inc (11/03/2015)
 15                              <1> ;
 16                              <1> ; Derived from 'IBM PC-AT' BIOS source code (1985)
 17                              <1> ; ****************************************************************************
 18                              <1>
 19                              <1> ; Retro UNIX 386 v1 Kernel - VIDATA.S
 20                              <1> ; Last Modification: 11/03/2015
 21                              <1> ;                (Data section for 'VIDEO.INC')
 22                              <1> ;
 23                              <1> ; ///////// VIDEO DATA ///////////////
 24                              <1>
 25                              <1> ;----------------------------------------
 26                              <1> ;     VIDEO DISPLAY DATA AREA          ;
 27                              <1> ;----------------------------------------
 28 00006FDA 03                  <1> CRT_MODE:    db     3      ; CURRENT DISPLAY MODE (TYPE)
 29 00006FDB 29                  <1> CRT_MODE_SET:      db     29h    ; CURRENT SETTING OF THE 3X8 REGISTER
 30                              <1>                            ; (29h default setting for video mode 3)
 31                              <1>                            ; Mode Select register Bits
 32                              <1>                            ;   BIT 0 - 80x25 (1), 40x25 (0)
 33                              <1>                            ;   BIT 1 - ALPHA (0), 320x200 GRAPHICS (1)
 34                              <1>                            ;   BIT 2 - COLOR (0), BW (1)
 35                              <1>                            ;   BIT 3 - Video Sig. ENABLE (1), DISABLE (0)
 36                              <1>                            ;   BIT 4 - 640x200 B&W Graphics Mode (1)
 37                              <1>                            ;   BIT 5 - ALPHA mode BLINKING (1)
 38                              <1>                            ;   BIT 6, 7 - Not Used
 39                              <1>
 40                              <1> ; Mode 0 - 2Ch = 101100b  ; 40x25 text, 16 gray colors
 41                              <1> ; Mode 1 - 28h = 101000b  ; 40x25 text, 16 fore colors, 8 back colors
 42                              <1> ; Mode 2 - 2Dh = 101101b  ; 80x25 text, 16 gray colors
 43                              <1> ; Mode 3 - 29h = 101001b  ; 80x25 text, 16 fore color, 8 back color
 44                              <1> ; Mode 4 - 2Ah = 101010b  ; 320x200 graphics, 4 colors
 45                              <1> ; Mode 5 - 2Eh = 101110b  ; 320x200 graphics, 4 gray colors
 46                              <1> ; Mode 6 - 1Eh = 011110b  ; 640x200 graphics, 2 colors
 47                              <1> ; Mode 7 - 29h = 101001b  ; 80x25 text, black & white colors
 48                              <1> ; Mode & 37h = Video signal OFF
 49                              <1>
 50                              <1> ; 24/06/2016
 51 00006FDC 50                  <1> CRT_COLS:    db     80     ; Number of columns
 52                              <1>
 53                              <1> ; 01/07/2016
 54 00006FDD 00                  <1> CRT_PALETTE: db     0      ; Current palette setting
 55                              <1>
 56                              <1> ; 03/07/2016
 57 00006FDE 10                  <1> CHAR_HEIGHT: db     16     ; Default character height
 58 00006FDF 60                  <1> VGA_VIDEO_CTL:     db     60h    ; ROM BIOS DATA AREA Offset 87h
 59 00006FE0 F9                  <1> VGA_SWITCHES:      db     0F9h   ; Feature Bit Switches (the basic screen)
 60 00006FE1 51                  <1> VGA_MODESET_CTL: db       051h   ; Basic mode set options (VGA video flags)
 61                              <1>                            ; ROM BIOS DATA AREA Offset 89h
 62                              <1>                            ; Bit 7, 4 : Mode
 63                              <1>                            ;       01 : 400-line mode
 64                              <1>                            ; Bit 6      : Display switch enabled = 1
 65                              <1>                            ; Bit 5      : Reserved = 0
 66                              <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
00006FE2 19                     <1> VGA_ROWS:   db     25
                                <1>
                                <1> ; 16/01/2016
                                <1> chr_attrib:  ; Character color/attributes for video pages (0 to 7)
00006FE3 0707070707070707       <1>       db     07h, 07h, 07h, 07h, 07h, 07h, 07h, 07h
                                <1> ; 30/01/2016
                                <1> vmode:
00006FEB 0303030303030303       <1>       db     3,3,3,3,3,3,3,3 ; video modes for pseudo screens
                                <1>
                                <1> CURSOR_MODE: ; cursor start (ch) = 14, cursor end (cl) = 15
00006FF3 0F0E                   <1>       db     15, 14 ; 07/07/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>
                                <1> ;align 4
                                <1> ;VGA_BASE: ; 26/07/2016
                                <1> ;     dd      0B8000h  ; (Mode < 0Dh) or 0A0000h (mode >= 0Dh)
                                <1>
00006FF5 90                     <1> align 2
                                <1>
                                <1> vga_modes:
                                <1>       ; 25/07/2016
                                <1>       ; 09/07/2016
                                <1>       ; 03/07/2016
                                <1>       ; valid (implemented) video modes (>7, extension to IBM PC CGA modes)
00006FF6 0302010007040506       <1>       db     03h, 02h, 01h, 00h, 07h, 04h, 05h, 06h
                                <1> vga_g_modes: ; 31/07/2016
00006FFE 13F0126A0D0E1011       <1>       db     13h, 0F0h, 12h, 6Ah, 0Dh, 0Eh, 10h, 11h
                                <1> vga_mode_count equ $ - vga_modes
                                <1> vga_g_mode_count equ $ - vga_g_modes
                                <1>
                                <1> vga_mode_tbl_ptr:
                                <1>       ; 25/07/2016
00007006 [66700000]             <1>       dd     vga_mode_03h
0000700A [66700000]             <1>       dd     vga_mode_03h ; mode 02h -> mode 03h
0000700E [A6700000]             <1>       dd     vga_mode_01h
00007012 [A6700000]             <1>       dd     vga_mode_01h ; mode 00h -> mode 01h
                                <1>       ;dd    vga_mode_07h
00007016 [66700000]             <1>       dd     vga_mode_03h ; mode 07h -> mode 03h
0000701A [E6700000]             <1>       dd     vga_mode_04h
0000701E [E6700000]             <1>       dd     vga_mode_04h ; mode 05h -> mode 04h
00007022 [26710000]             <1>       dd     vga_mode_06h
00007026 [66710000]             <1>       dd     vga_mode_13h
0000702A [A6710000]             <1>       dd     vga_mode_F0h
0000702E [E6710000]             <1>       dd     vga_mode_12h
00007032 [26720000]             <1>       dd     vga_mode_6Ah
00007036 [66720000]             <1>       dd     vga_mode_0Dh
0000703A [A6720000]             <1>       dd     vga_mode_0Eh
0000703E [E6720000]             <1>       dd     vga_mode_10h
00007042 [26730000]             <1>       dd     vga_mode_11h
                                <1>
                                <1> vga_memmodel:
                                <1>       ; 25/07/2016
                                <1>       ; 07/07/2016
                                <1>       CTEXT  equ 0
                                <1>       ;MTEXT equ 1
                                <1>       MTEXT  equ 0 ; mode 07h -> mode 03h
                                <1>       CGA    equ 2
                                <1>       LINEAR8 equ 5
                                <1>       PLANAR4      equ 4
                                <1>       PLANAR1      equ 3
00007046 0000000000020202       <1>       db     CTEXT, CTEXT, CTEXT, CTEXT, MTEXT, CGA, CGA, CGA
                                <1> vga_g_memmodel: ; 31/07/2016
0000704E 0504040404040403       <1>       db     LINEAR8, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR4, PLANAR1
                                <1> ;vga_pixbits:
                                <1> ;     ; 25/07/2016
                                <1> ;     ; 08/07/2016
                                <1> ;     db     4, 4, 4, 4, 4, 2, 2, 1, 8, 4, 4, 4, 4, 4, 4, 1
                                <1> vga_dac_s:
00007056 020202020001010103-    <1>       db     2, 2, 2, 2, 0, 1, 1, 1, 3, 3, 2, 2, 1, 1, 2, 2
0000705F 03020201010202         <1>
                                <1>                                     ; (vgatables.h, VGAMODES, dac)
                                <1>                                     ; 17/11/2020
                                <1> vga_params:
                                <1>       ; 23/11/2020
                                <1>       ; 16/11/2020
                                <1>       ; 09/11/2020, 10/11/2020, 11/11/2020 (TRDOS 386 v2.0.3)
                                <1>       ; 25/07/2016
                                <1>       ; 19/07/2016
                                <1>       ; 03/07/2016
                                <1>       ; derived from 'Plex86/Bochs VGABios' source code
                                <1>       ; vgabios-0.7a (2011)
                                <1>       ; by the LGPL VGABios Developers Team (2001-2008)
                                <1>       ; 'vgatables.h'
                                <1>       ; Oracle VirtualBox 5.0.24 VGABios Source Code
                                <1>       ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
                                <1>
                                <1>       ; 09/11/2020
                                <1>       ; Block Structure of Video Parameter Table
                                <1>       ;
                                <1>       ; Offset     # Bytes             Contents
                                <1>       ;
                                <1>       ;  0         1            # columns
                                <1>       ;  1         1            # rows - 1
                                <1>       ;  2         1            Pixels/character
                                <1>       ;  3-4       2            Page length
                                <1>       ;  5-8       4            Sequencer Registers
                                <1>       ;  9         1            Miscellaneous Register
                                <1>       ;  10-34     25           CRTC Registers
                                <1>       ;  35-54     20           Attribute Registers
                                <1>       ;  55-63     9            Graphics Controller Registers
                                <1>       ;
```

```
                                <1>       ; Ref: Programmer's Guide to EGA, VGA, and Super VGA cards
                                <1>       ; (Richard F. Ferraro, 1994)
                                <1>
                                <1>       ;
                                <1> vga_mode_03h:  ; mode 03h, 80*25 text, CGA colors
                                <1>       ; 11/11/2020
00007066 5018100010             <1>       db     80, 24, 16, 00h, 10h ; tw, th-1, ch, slength (5)
0000706B 00030002               <1>       db     00h, 03h, 00h, 02h ; sequ regs (4)
0000706F 67                     <1>       db     67h    ; misc reg (1)
00007070 5F4F50825581BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
                                <1>       ; 09/11/2020
                                <1>       ;db    5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007078 004F                   <1>       db     00h, 4Fh
                                <1> vga_p_cm_pos equ $ - vga_mode_03h
0000707A 0D0E00000000           <1>       db     0Dh, 0Eh, 00h, 00h, 00h, 00h
00007080 9C8E8F281F96B9A3       <1>       db     9Ch, 8Eh, 8Fh, 28h, 1Fh, 96h, 0B9h, 0A3h
00007088 FF                     <1>       db     0FFh   ; crtc_regs (25)
00007089 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007091 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
                                <1>       ;db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
00007099 0C000F00               <1>       db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
0000709D 0000000000100E0FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs (9)
                                <1>       ; 09/11/2020
                                <1>       ;db    00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 00h, 0FFh ; grdc regs (9)
                                <1> vga_mode_01h:      ; mode 01h, 40*25 text, CGA colors
000070A6 2818100008             <1>       db     40, 24, 16, 00h, 08h ; tw, th-1, ch, slength
000070AB 08030002               <1>       db     08h, 03h, 00h, 02h  ; sequ regs
000070AF 67                     <1>       db     67h    ; misc reg
000070B0 2D2728902BA0BF1F       <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 0A0h, 0BFh, 1Fh
000070B8 004F0D0E00000000       <1>       db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
000070C0 9C8E8F141F96B9A3       <1>       db     9Ch, 8Eh, 8Fh, 14h, 1Fh, 96h, 0B9h, 0A3h
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
                                <1>       ;db    0Ch, 00h, 0Fh, 08h  ; actl regs (20)
000070D9 0C000F00               <1>       db     0Ch, 00h, 0Fh, 00h  ; 19/11/2020
000070DD 0000000000100E0FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 10h, 0Eh, 0Fh, 0FFh ; grdc regs
                                <1> ;vga_mode_07h:     ; mode 07h, 80*25 text, mono color
                                <1> ;     db     80, 24, 16, 00h, 10h  ; tw, th-1, ch, slength
                                <1> ;     db     00h, 03h, 00h, 02h ; sequ regs
                                <1> ;     db     66h    ; misc reg
                                <1> ;     db     5Fh, 4Fh, 50h, 82h, 55h, 81h, 0BFh, 1Fh
                                <1> ;     db     00h, 4Fh, 0Dh, 0Eh, 00h, 00h, 00h, 00h
                                <1> ;     db     9Ch, 8Eh, 8Fh, 28h, 0Fh, 96h, 0B9h, 0A3h
                                <1> ;     db     0FFh   ; crtc regs
                                <1> ;     db     00h, 08h, 08h, 08h, 08h, 08h, 08h, 08h
                                <1> ;     db     10h, 18h, 18h, 18h, 18h, 18h, 18h, 18h
                                <1> ;     db     0Eh, 00h, 0Fh, 08h  ; actl regs
                                <1> ;     db     00h, 00h, 00h, 00h, 00h, 10h, 0Ah, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_04h:      ; 320*200 graphics, 4 colors, CGA
                                <1>       ; 11/11/2020
000070E6 2818080040             <1>       db     40, 24, 8, 00h, 40h ; tw, th-1, ch, slength
000070EB 09030002               <1>       db     09h, 03h, 00h, 02h ; sequ regs
000070EF 63                     <1>       db     63h    ; misc reg
000070F0 2D2728902B80BF1F       <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
000070F8 00C1000000000000       <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
00007100 9C8E8F140096B9A2       <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0A2h
00007108 FF                     <1>       db     0FFh   ; crtc_regs
00007109 0013151702040607       <1>       db     00h, 13h, 15h, 17h, 02h, 04h, 06h, 07h
00007111 1011121314151617       <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00007119 01000300               <1>       db     01h, 00h, 03h, 00h ; actl regs
0000711D 0000000000300F0FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 30h, 0Fh, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_06h:      ; 640*200 graphics, 2 colors, CGA
                                <1>       ; 11/11/2020
00007126 5018080040             <1>       db     80, 24, 8, 00h, 40h   ;   tw, th-1, ch, slength
0000712B 01010006               <1>       db     01h, 01h, 00h, 06h ; sequ regs
0000712F 63                     <1>       db     63h    ; misc reg
00007130 5F4F50825480BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007138 00C1000000000000       <1>       db     00h, 0C1h, 00h, 00h, 00h, 00h, 00h, 00h
00007140 9C8E8F280096B9C2       <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0C2h
00007148 FF                     <1>       db     0FFh   ; crtc regs
00007149 0017171717171717       <1>       db     00h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
00007151 1717171717171717       <1>       db     17h, 17h, 17h, 17h, 17h, 17h, 17h, 17h
00007159 01000100               <1>       db     01h, 00h, 01, 00h  ; actl regs
0000715D 0000000000000D0FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 00h, 0Dh, 0Fh, 0FFh ; grdc regs
                                <1> vga_mode_13h:  ; mode 13h, 300*200, 256 colors, linear
                                <1>       ; 11/11/2020
                                <1>       ;db    40, 24, 8, 00h, 20h ; tw, th-1, ch, slength (5)
                                <1>       ; 23/11/2020 - 10/11/2020
00007166 28180800FA             <1>       db     40, 24, 8, 00h, 0FAh ; tw, th-1, ch, slength (5)
0000716B 010F000E               <1>       db     01h, 0Fh, 00h, 0Eh ; sequ regs (4)
0000716F 63                     <1>       db     63h    ; misc reg (1)
00007170 5F4F50825480BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
00007178 0041000000000000       <1>       db     00h, 041h, 00h, 00h, 00h, 00h, 00h, 00h
00007180 9C8E8F284096B9A3       <1>       db     9Ch, 8Eh, 8Fh, 28h, 40h, 96h, 0B9h, 0A3h
00007188 FF                     <1>       db     0FFh   ; crtc regs (25)
00007189 0001020304050607       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00007191 08090A0B0C0D0E0F       <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
00007199 41000F00               <1>       db     41h, 00h, 0Fh, 00h  ; actl regs (20)
                                <1>       ; 10/11/2020
                                <1>       ;db    41h, 01h, 0Fh, 13h  ; actl regs (20)
0000719D 000000000040050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs (9)
                                <1> vga_mode_setl equ $ - vga_mode_13h  ; = 64
                                <1> vga_mode_F0h:  ; mode X ; 320*240, 256 colors, planar
000071A6 2818080000             <1>       db     40, 24, 8, 00h, 00h ; tw, th-1, ch, slength
000071AB 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000071AF E3                     <1>       db     0E3h   ; misc reg
000071B0 5F4F508254800D3E       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Dh, 3Eh
000071B8 0041000000000000       <1>       db     00h, 41h, 00h, 00h, 00h, 00h, 00h, 00h
000071C0 EAACDF2800E706E3       <1>       db     0EAh, 0ACh, 0DFh, 28h, 00h, 0E7h, 06h, 0E3h
000071C8 FF                     <1>       db     0FFh   ; crtc regs (25)
000071C9 0001020304050607       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
000071D1 08090A0B0C0D0E0F       <1>       db     08h, 09h, 0Ah, 0Bh, 0Ch, 0Dh, 0Eh, 0Fh
000071D9 41000F00               <1>       db     41h, 00h, 0Fh, 00h  ; actl regs
000071DD 000000000040050FFF     <1>       db     00h, 00h, 00h, 00h, 00h, 40h, 05h, 0Fh, 0FFh ; grdc regs
```

```
                                <1> vga_mode_12h:  ; mode 12h, 640*480, 16 colors, planar
                                <1>       ; 11/11/2020
                                <1>       ;db    80, 29, 16, 0, 0 ; tw, th-1, ch, slength
                                <1>       ; 09/11/2020
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
                                <1>       ; 09/11/2020
                                <1>       ;db    5Fh, 4Fh, 50h, 82h, 53h, 9Fh, 0Bh, 3Eh
000071F8 0040000000000000       <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00007200 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
00007208 FF                     <1>       db     0FFh   ; crtc regs
00007209 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007211 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 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> vga_mode_6Ah:  ; mode 6Ah, 800*600, 16 colors, planar
                                <1>       ; 11/11/2020
00007226 6424100000             <1>       db     100, 36, 16, 00h, 00h ; tw, th-1, ch, slength
0000722B 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000722F E3                     <1>       db     0E3h   ; misc reg
00007230 7F6363836B1B72F0       <1>       db     7Fh, 63h, 63h, 83h, 6Bh, 1Bh, 72h, 0F0h
00007238 0060000000000000       <1>       db     00h, 60h, 00h, 00h, 00h, 00h, 00h, 00h
00007240 598D5732005773E3       <1>       db     59h, 8Dh, 57h, 32h, 00h, 57h, 73h, 0E3h
00007248 FF                     <1>       db     0FFh   ; crtc regs
00007249 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007251 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00007259 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000725D 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
00007266 2818080020             <1>       db     40, 24, 8, 00h, 20h ; tw, th-1, ch, slength
0000726B 090F0006               <1>       db     09h, 0Fh, 00h, 06h ; sequ regs
0000726F 63                     <1>       db     63h    ; misc reg
00007270 2D2728902B80BF1F       <1>       db     2Dh, 27h, 28h, 90h, 2Bh, 80h, 0BFh, 1Fh
00007278 00C0000000000000       <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
00007280 9C8E8F140096B9E3       <1>       db     9Ch, 8Eh, 8Fh, 14h, 00h, 96h, 0B9h, 0E3h
00007288 FF                     <1>       db     0FFh   ; crtc regs
00007289 0001020304050607       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
00007291 1011121314151617       <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
00007299 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000729D 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
000072A6 5018080040             <1>       db     80, 24, 8, 00h, 40h ; tw, th-1, ch, slength
000072AB 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000072AF 63                     <1>       db     63h    ; misc reg
000072B0 5F4F50825480BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
000072B8 00C0000000000000       <1>       db     00h, 0C0h, 00h, 00h, 00h, 00h, 00h, 00h
000072C0 9C8E8F280096B9E3       <1>       db     9Ch, 8Eh, 8Fh, 28h, 00h, 96h, 0B9h, 0E3h
000072C8 FF                     <1>       db     0FFh   ; crtc regs
000072C9 0001020304050607       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 06h, 07h
000072D1 1011121314151617       <1>       db     10h, 11h, 12h, 13h, 14h, 15h, 16h, 17h
000072D9 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
000072DD 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
000072E6 50180E0080             <1>       db     80, 24, 14, 00h, 80h ; tw, th-1, ch, slength
000072EB 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
000072EF A3                     <1>       db     0A3h   ; misc reg
000072F0 5F4F50825480BF1F       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0BFh, 1Fh
000072F8 0040000000000000       <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00007300 83855D280F63BAE3       <1>       db     83h, 85h, 5Dh, 28h, 0Fh, 63h, 0BAh, 0E3h
00007308 FF                     <1>       db     0FFh   ; crtc regs
00007309 0001020304051407       <1>       db     00h, 01h, 02h, 03h, 04h, 05h, 14h, 07h
00007311 38393A3B3C3D3E3F       <1>       db     38h, 39h, 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh
00007319 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000731D 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
00007326 501D1000A0             <1>       db     80, 29, 16, 00h, 0A0h ; tw, th-1, ch, slength
0000732B 010F0006               <1>       db     01h, 0Fh, 00h, 06h ; sequ regs
0000732F E3                     <1>       db     0E3h   ; misc reg
00007330 5F4F508254800B3E       <1>       db     5Fh, 4Fh, 50h, 82h, 54h, 80h, 0Bh, 3Eh
00007338 0040000000000000       <1>       db     00h, 40h, 00h, 00h, 00h, 00h, 00h, 00h
00007340 EA8CDF2800E704C3       <1>       db     0EAh, 8Ch, 0DFh, 28h, 00h, 0E7h, 04h, 0C3h ; 11/11/2020
00007348 FF                     <1>       db     0FFh   ; crtc regs
00007349 003F003F003F003F       <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00007351 003F003F003F003F       <1>       db     00h, 3Fh, 00h, 3Fh, 00h, 3Fh, 00h, 3Fh
00007359 01000F00               <1>       db     01h, 00h, 0Fh, 00h  ; actl regs
0000735D 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 */
00007366 0001800290010800       <1>       dw 100h,  640, 400,  8
0000736E 01018002E0010800       <1>       dw 101h,  640, 480,  8
00007376 0301200358020800       <1>       dw 103h,  800, 600,  8
0000737E 0501000400030800       <1>       dw 105h, 1024, 768,  8
00007386 0E014001C8001000       <1>       dw 10Eh,  320, 200, 16
0000738E 0F014001C8001800       <1>       dw 10Fh,  320, 200, 24
00007396 11018002E0011000       <1>       dw 111h,  640, 480, 16
0000739E 12018002E0011800       <1>       dw 112h,  640, 480, 24
000073A6 1401200358021000       <1>       dw 114h,  800, 600, 16
000073AE 1501200358021800       <1>       dw 115h,  800, 600, 24
000073B6 1701000400031000       <1>       dw 117h, 1024, 768, 16
000073BE 1801000400031800       <1>       dw 118h, 1024, 768, 24
```

```
                                <1>
                                <1> ;/* BOCHS/PLEX86 'own' mode numbers */
000073C6 40014001C8002000       <1>       dw 140h,  320,  200, 32
000073CE 4101800290012000       <1>       dw 141h,  640,  400, 32
000073D6 42018002E0012000       <1>       dw 142h,  640,  480, 32
000073DE 4301200358022000       <1>       dw 143h,  800,  600, 32
000073E6 4401000400032000       <1>       dw 144h, 1024,  768, 32
000073EE 46014001C8000800       <1>       dw 146h,  320,  200,  8
000073F6 8D010005D0021000       <1>       dw 18Dh, 1280,  720, 16
000073FE 8E010005D0021800       <1>       dw 18Eh, 1280,  720, 24
00007406 8F010005D0022000       <1>       dw 18Fh, 1280,  720, 32
0000740E 9001800738041000       <1>       dw 190h, 1920, 1080, 16
00007416 9101800738041800       <1>       dw 191h, 1920, 1080, 24
0000741E 9201800738042000       <1>       dw 192h, 1920, 1080, 32
                                <1>
                                <1> end_of_b_vbe_modes:
                                <1>
                                <1> MA1 equ VBE_MODE_ATTRIBUTE_SUPPORTED
                                <1> MA2 equ VBE_MODE_ATTRIBUTE_EXTENDED_INFO_AVAILABLE
                                <1> MA3 equ VBE_MODE_ATTRIBUTE_COLOR_MODE
                                <1> MA4 equ VBE_MODE_ATTRIBUTE_LINEAR_FRAME_BUFFER_MODE
                                <1> MA5 equ     VBE_MODE_ATTRIBUTE_GRAPHICS_MODE
                                <1>
                                <1> MODE_ATTRIBUTES equ MA1|MA2|MA3|MA4|MA5
                                <1>
                                <1> WA1 equ VBE_WINDOW_ATTRIBUTE_RELOCATABLE
                                <1> WA2 equ     VBE_WINDOW_ATTRIBUTE_READABLE
                                <1> WA3 equ VBE_WINDOW_ATTRIBUTE_WRITEABLE
                                <1>
                                <1> WINA_ATTRIBUTES equ WA1|WA2|WA3
                                <1>
                                <1> ; 24/11/2020
                                <1>
                                <1> %if 0
                                <1>
                                <1> MODE_INFO_LIST:
                                <1>
                                <1> ; 24/11/2020
                                <1> ; '%if 0' disables 24 mode info tables here, until %endif
                                <1> ; ('set_mode_info_list' will set only 1 list for selected mode)
                                <1> ; (Purpose: To save about 1 KB kernel size by removing fixed data)
                                <1>
                                <1>                     dw     0100h ; 640x400x8
                                <1> ModeAttributes1:    dw     MODE_ATTRIBUTES
                                <1> WinAAttributes1:    db     WINA_ATTRIBUTES
                                <1> WinBAttributes1:    db     0
                                <1> WinGranularity1:    dw     VBE_DISPI_BANK_SIZE_KB
                                <1> WinSize1:           dw     VBE_DISPI_BANK_SIZE_KB
                                <1> WinASegment1:               dw     VGAMEM_GRAPH
                                <1> WinBSegment1:               dw     0000h
                                <1> WinFuncPtr1:        dd     0
                                <1> BytesPerScanLine1:          dw     640
                                <1> XResolution1:               dw     640
                                <1> YResolution1:               dw     400
                                <1> XCharSize1:         db     8
                                <1> YCharSize1:         db     16
                                <1> NumberOfPlanes1:    db     1
                                <1> BitsPerPixel1:              db     8
                                <1> NumberOfBanks1:             db     4
                                <1> MemoryModel1:               db     VBE_MEMORYMODEL_PACKED_PIXEL
                                <1> BankSize1:          db     0
                                <1> NumberOfImagePages1:        db     64
                                <1> Reserved_page1:             db     0
                                <1> RedMaskSize1:               db     0
                                <1> RedFieldPosition1:          db     0
                                <1> GreenMaskSize1:             db     0
                                <1> GreenFieldPosition1:        db     0
                                <1> BlueMaskSize1:              db     0
                                <1> BlueFieldPosition1:         db     0
                                <1> RsvdMaskSize1:              db     0
                                <1> RsvdFieldPosition1:         db     0
                                <1> DirectColorModeInfo1:       db     0
                                <1> PhysBasePtr1:               dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
                                <1> OffScreenMemOffset1:        dd     0
                                <1> OffScreenMemSize1:          dw     0
                                <1> LinBytesPerScanLine1:       dw     640
                                <1> BnkNumberOfPages1:          db     0
                                <1> LinNumberOfPages1:          db     0
                                <1> LinRedMaskSize1:    db     0
                                <1> LinRedFieldPosition1:       db     0
                                <1> LinGreenMaskSize1:          db     0
                                <1> LinGreenFieldPosition1: db      0
                                <1> LinBlueMaskSize1:   db     0
                                <1> LinBlueFieldPosition1:      db     0
                                <1> LinRsvdMaskSize1:   db     0
                                <1> LinRsvdFieldPosition1:      db     0
                                <1> MaxPixelClock1:             dd     0
                                <1>
                                <1>                     dw     0101h ; 640x480x8
                                <1> ModeAttributes2:    dw     MODE_ATTRIBUTES
                                <1> WinAAttributes2:    db     WINA_ATTRIBUTES
                                <1> WinBAttributes2:    db     0
                                <1> WinGranularity2:    dw     VBE_DISPI_BANK_SIZE_KB
                                <1> WinSize2:           dw     VBE_DISPI_BANK_SIZE_KB
                                <1> WinASegment2:               dw     VGAMEM_GRAPH
                                <1> WinBSegment2:               dw     0000h
                                <1> WinFuncPtr2:        dd     0
                                <1> BytesPerScanLine2: dw      640
                                <1> XResolution2:               dw     640
                                <1> YResolution2:               dw     480
                                <1> XCharSize2:         db     8
                                <1> YCharSize2:         db     16
                                <1> NumberOfPlanes2:    db     1
                                <1> BitsPerPixel2:              db     8
                                <1> NumberOfBanks2:             db     5
```

```
                        <1> MemoryModel2:             db    VBE_MEMORYMODEL_PACKED_PIXEL
                        <1> BankSize2:          db    0
                        <1> NumberOfImagePages2:      db    53
                        <1> Reserved_page2:           db    0
                        <1> RedMaskSize2:             db    0
                        <1> RedFieldPosition2: db     0
                        <1> GreenMaskSize2:           db    0
                        <1> GreenFieldPosition2:      db    0
                        <1> BlueMaskSize2:            db    0
                        <1> BlueFieldPosition2:       db    0
                        <1> RsvdMaskSize2:            db    0
                        <1> RsvdFieldPosition2:       db    0
                        <1> DirectColorModeInfo2:     db    0
                        <1> PhysBasePtr2:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset2:      dd    0
                        <1> OffScreenMemSize2: dw     0
                        <1> LinBytesPerScanLine2:     dw    640
                        <1> BnkNumberOfPages2:        db    0
                        <1> LinNumberOfPages2:        db    0
                        <1> LinRedMaskSize2:   db     0
                        <1> LinRedFieldPosition2:     db    0
                        <1> LinGreenMaskSize2:        db    0
                        <1> LinGreenFieldPosition2: db        0
                        <1> LinBlueMaskSize2:  db     0
                        <1> LinBlueFieldPosition2:    db    0
                        <1> LinRsvdMaskSize2:  db     0
                        <1> LinRsvdFieldPosition2:    db    0
                        <1> MaxPixelClock2:           dd    0
                        <1>
                        <1>                     dw    0103h ; 800x600x8
                        <1> ModeAttributes3:    dw    MODE_ATTRIBUTES
                        <1> WinAAttributes3:    db    WINA_ATTRIBUTES
                        <1> WinBAttributes3:    db    0
                        <1> WinGranularity3:    dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize3:           dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment3:             dw    VGAMEM_GRAPH
                        <1> WinBSegment3:             dw    0000h
                        <1> WinFuncPtr3:        dd    0
                        <1> BytesPerScanLine3: dw     800
                        <1> XResolution3:             dw    800
                        <1> YResolution3:             dw    600
                        <1> XCharSize3:         db    8
                        <1> YCharSize3:         db    16
                        <1> NumberOfPlanes3:    db    1
                        <1> BitsPerPixel3:            db    8
                        <1> NumberOfBanks3:           db    8
                        <1> MemoryModel3:             db    VBE_MEMORYMODEL_PACKED_PIXEL
                        <1> BankSize3:          db    0
                        <1> NumberOfImagePages3:      db    33
                        <1> Reserved_page3:           db    0
                        <1> RedMaskSize3:             db    0
                        <1> RedFieldPosition3: db     0
                        <1> GreenMaskSize3:           db    0
                        <1> GreenFieldPosition3:      db    0
                        <1> BlueMaskSize3:            db    0
                        <1> BlueFieldPosition3:       db    0
                        <1> RsvdMaskSize3:            db    0
                        <1> RsvdFieldPosition3:       db    0
                        <1> DirectColorModeInfo3:     db    0
                        <1> PhysBasePtr3:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset3:      dd    0
                        <1> OffScreenMemSize3: dw     0
                        <1> LinBytesPerScanLine3:     dw    800
                        <1> BnkNumberOfPages3:        db    0
                        <1> LinNumberOfPages3:        db    0
                        <1> LinRedMaskSize3:   db     0
                        <1> LinRedFieldPosition3:     db    0
                        <1> LinGreenMaskSize3:        db    0
                        <1> LinGreenFieldPosition:    db    0
                        <1> LinBlueMaskSize:   db     0
                        <1> LinBlueFieldPosition:     db    0
                        <1> LinRsvdMaskSize:   db     0
                        <1> LinRsvdFieldPosition:     db    0
                        <1> MaxPixelClock:            dd    0
                        <1>
                        <1>                     dw    0105h ; 1024x768x8
                        <1> ModeAttributes4:    dw    MODE_ATTRIBUTES
                        <1> WinAAttributes4:    db    WINA_ATTRIBUTES
                        <1> WinBAttributes4:    db    0
                        <1> WinGranularity4:    dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize4:           dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment4:             dw    VGAMEM_GRAPH
                        <1> WinBSegment4:             dw    0000h
                        <1> WinFuncPtr4:        dd    0
                        <1> BytesPerScanLine4: dw     1024
                        <1> XResolution4:             dw    1024
                        <1> YResolution4:             dw    768
                        <1> XCharSize4:         db    8
                        <1> YCharSize4:         db    16
                        <1> NumberOfPlanes4:    db    1
                        <1> BitsPerPixel4:            db    8
                        <1> NumberOfBanks4:           db    12
                        <1> MemoryModel4:             db    VBE_MEMORYMODEL_PACKED_PIXEL
                        <1> BankSize4:          db    0
                        <1> NumberOfImagePages4:      db    20
                        <1> Reserved_page4:           db    0
                        <1> RedMaskSize4:             db    0
                        <1> RedFieldPosition4: db     0
                        <1> GreenMaskSize4:           db    0
                        <1> GreenFieldPosition4:      db    0
                        <1> BlueMaskSize4:            db    0
                        <1> BlueFieldPosition4:       db    0
                        <1> RsvdMaskSize4:            db    0
                        <1> RsvdFieldPosition4:       db    0
                        <1> DirectColorModeInfo4:     db    0
```

```
591                                     <1> PhysBasePtr4:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
592                                     <1> OffScreenMemOffset4:      dd    0
593                                     <1> OffScreenMemSize4: dw     0
594                                     <1> LinBytesPerScanLine4:     dw    1024
595                                     <1> BnkNumberOfPages4:        db    0
596                                     <1> LinNumberOfPages4:        db    0
597                                     <1> LinRedMaskSize4:   db     0
598                                     <1> LinRedFieldPosition4:     db    0
599                                     <1> LinGreenMaskSize4:        db    0
600                                     <1> LinGreenFieldPosition4: db      0
601                                     <1> LinBlueMaskSize4:  db     0
602                                     <1> LinBlueFieldPosition4:    db    0
603                                     <1> LinRsvdMaskSize4:  db     0
604                                     <1> LinRsvdFieldPosition4:    db    0
605                                     <1> MaxPixelClock4:           dd    0
606                                     <1>
607                                     <1>                   dw    010Eh ; 320x200x16
608                                     <1> ModeAttributes5:  dw    MODE_ATTRIBUTES
609                                     <1> WinAAttributes5:  db    WINA_ATTRIBUTES
610                                     <1> WinBAttributes5:  db    0
611                                     <1> WinGranularity5:  dw    VBE_DISPI_BANK_SIZE_KB
612                                     <1> WinSize5:         dw    VBE_DISPI_BANK_SIZE_KB
613                                     <1> WinASegment5:             dw    VGAMEM_GRAPH
614                                     <1> WinBSegment5:             dw    0000h
615                                     <1> WinFuncPtr5:      dd    0
616                                     <1> BytesPerScanLine5: dw     640
617                                     <1> XResolution5:             dw    320
618                                     <1> YResolution5:             dw    200
619                                     <1> XCharSize5:       db    8
620                                     <1> YCharSize5:       db    16
621                                     <1> NumberOfPlanes5:  db    1
622                                     <1> BitsPerPixel5:            db    16
623                                     <1> NumberOfBanks5:           db    2
624                                     <1> MemoryModel5:             db    VBE_MEMORYMODEL_DIRECT_COLOR
625                                     <1> BankSize5:        db    0
626                                     <1> NumberOfImagePages5:      db    130
627                                     <1> Reserved_page5:           db    0
628                                     <1> RedMaskSize5:             db    5
629                                     <1> RedFieldPosition5: db     11
630                                     <1> GreenMaskSize5:           db    6
631                                     <1> GreenFieldPosition5:      db    5
632                                     <1> BlueMaskSize5:            db    5
633                                     <1> BlueFieldPosition5:       db    0
634                                     <1> RsvdMaskSize5:            db    0
635                                     <1> RsvdFieldPosition5:       db    0
636                                     <1> DirectColorModeInfo5:     db    0
637                                     <1> PhysBasePtr5:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
638                                     <1> OffScreenMemOffset5:      dd    0
639                                     <1> OffScreenMemSize5: dw     0
640                                     <1> LinBytesPerScanLine5:     dw    640
641                                     <1> BnkNumberOfPages5:        db    0
642                                     <1> LinNumberOfPages5:        db    0
643                                     <1> LinRedMaskSize5:   db     5
644                                     <1> LinRedFieldPosition5:     db    11
645                                     <1> LinGreenMaskSize5: db     6
646                                     <1> LinGreenFieldPosition5:   db    5
647                                     <1> LinBlueMaskSize5:  db     5
648                                     <1> LinBlueFieldPosition5:    db    0
649                                     <1> LinRsvdMaskSize5:  db     0
650                                     <1> LinRsvdFieldPosition5:    db    0
651                                     <1> MaxPixelClock5:           dd    0
652                                     <1>
653                                     <1>                   dw    010Fh ; 320x200x24
654                                     <1> ModeAttributes6:  dw    MODE_ATTRIBUTES
655                                     <1> WinAAttributes6:  db    WINA_ATTRIBUTES
656                                     <1> WinBAttributes6:  db    0
657                                     <1> WinGranularity6:  dw    VBE_DISPI_BANK_SIZE_KB
658                                     <1> WinSize6:         dw    VBE_DISPI_BANK_SIZE_KB
659                                     <1> WinASegment6:             dw    VGAMEM_GRAPH
660                                     <1> WinBSegment6:             dw    0000h
661                                     <1> WinFuncPtr6:      dd    0
662                                     <1> BytesPerScanLine6: dw     960
663                                     <1> XResolution6:             dw    320
664                                     <1> YResolution6:             dw    200
665                                     <1> XCharSize6:       db    8
666                                     <1> YCharSize6:       db    16
667                                     <1> NumberOfPlanes6:  db    1
668                                     <1> BitsPerPixel6:            db    24
669                                     <1> NumberOfBanks6:           db    3
670                                     <1> MemoryModel6:             db    VBE_MEMORYMODEL_DIRECT_COLOR
671                                     <1> BankSize6:        db    0
672                                     <1> NumberOfImagePages6:      db    86
673                                     <1> Reserved_page6:           db    0
674                                     <1> RedMaskSize6:             db    8
675                                     <1> RedFieldPosition6: db     16
676                                     <1> GreenMaskSize6:           db    8
677                                     <1> GreenFieldPosition6:      db    8
678                                     <1> BlueMaskSize6:            db    8
679                                     <1> BlueFieldPosition6:       db    0
680                                     <1> RsvdMaskSize6:            db    0
681                                     <1> RsvdFieldPosition6:       db    0
682                                     <1> DirectColorModeInfo6:     db    0
683                                     <1> PhysBasePtr6:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
684                                     <1> OffScreenMemOffset6:      dd    0
685                                     <1> OffScreenMemSize6: dw     0
686                                     <1> LinBytesPerScanLine6:     dw    960
687                                     <1> BnkNumberOfPages6:        db    0
688                                     <1> LinNumberOfPages6:        db    0
689                                     <1> LinRedMaskSize6:   db     8
690                                     <1> LinRedFieldPosition6:     db    16
691                                     <1> LinGreenMaskSize6: db     8
692                                     <1> LinGreenFieldPosition6:   db    8
693                                     <1> LinBlueMaskSize6:  db     8
694                                     <1> LinBlueFieldPosition6:    db    0
695                                     <1> LinRsvdMaskSize6:  db     0
```

```
696                                     <1> LinRsvdFieldPosition6:    db     0
697                                     <1> MaxPixelClock6:           dd     0
698                                     <1>
699                                     <1>                       dw     0111h ; 640x480x16
700                                     <1> ModeAttributes7:      dw     MODE_ATTRIBUTES
701                                     <1> WinAAttributes7:      db     WINA_ATTRIBUTES
702                                     <1> WinBAttributes7:      db     0
703                                     <1> WinGranularity7:      dw     VBE_DISPI_BANK_SIZE_KB
704                                     <1> WinSize7:             dw     VBE_DISPI_BANK_SIZE_KB
705                                     <1> WinASegment7:                dw     VGAMEM_GRAPH
706                                     <1> WinBSegment7:                dw     0000h
707                                     <1> WinFuncPtr7:          dd     0
708                                     <1> BytesPerScanLine7:    dw     1280
709                                     <1> XResolution7:                dw     640
710                                     <1> YResolution7:                dw     480
711                                     <1> XCharSize7:           db     8
712                                     <1> YCharSize7:           db     16
713                                     <1> NumberOfPlanes7:      db     1
714                                     <1> BitsPerPixel7:               db     16
715                                     <1> NumberOfBanks7:              db     10
716                                     <1> MemoryModel7:                db     VBE_MEMORYMODEL_DIRECT_COLOR
717                                     <1> BankSize7:            db     0
718                                     <1> NumberOfImagePages7:         db     26
719                                     <1> Reserved_page7:              db     0
720                                     <1> RedMaskSize7:                db     5
721                                     <1> RedFieldPosition7:    db     11
722                                     <1> GreenMaskSize7:              db     6
723                                     <1> GreenFieldPosition7:         db     5
724                                     <1> BlueMaskSize7:               db     5
725                                     <1> BlueFieldPosition7:          db     0
726                                     <1> RsvdMaskSize7:               db     0
727                                     <1> RsvdFieldPosition7:          db     0
728                                     <1> DirectColorModeInfo7:        db     0
729                                     <1> PhysBasePtr7:                dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS,
730                                     <1> OffScreenMemOffset7:         dd     0
731                                     <1> OffScreenMemSize7:    dw     0
732                                     <1> LinBytesPerScanLine7:        dw     1280
733                                     <1> BnkNumberOfPages7:           db     0
734                                     <1> LinNumberOfPages7:           db     0
735                                     <1> LinRedMaskSize7:      db     5
736                                     <1> LinRedFieldPosition7:        db     11
737                                     <1> LinGreenMaskSize7:    db     6
738                                     <1> LinGreenFieldPosition7:      db     5
739                                     <1> LinBlueMaskSize7:     db     5
740                                     <1> LinBlueFieldPosition7:       db     0
741                                     <1> LinRsvdMaskSize7:     db     0
742                                     <1> LinRsvdFieldPosition7:       db     0
743                                     <1> MaxPixelClock7:              dd     0
744                                     <1>
745                                     <1>                       dw     0112h ; 640x480x24
746                                     <1> ModeAttributes8:      dw     MODE_ATTRIBUTES
747                                     <1> WinAAttributes8:      db     WINA_ATTRIBUTES
748                                     <1> WinBAttributes8:      db     0
749                                     <1> WinGranularity8:      dw     VBE_DISPI_BANK_SIZE_KB
750                                     <1> WinSize8:             dw     VBE_DISPI_BANK_SIZE_KB
751                                     <1> WinASegment8:                dw     VGAMEM_GRAPH
752                                     <1> WinBSegment8:                dw     0000h
753                                     <1> WinFuncPtr8:          dd     0
754                                     <1> BytesPerScanLine8:    dw     1920
755                                     <1> XResolution8:                dw     640
756                                     <1> YResolution8:                dw     480
757                                     <1> XCharSize8:           db     8
758                                     <1> YCharSize8:           db     16
759                                     <1> NumberOfPlanes8:      db     1
760                                     <1> BitsPerPixel8:               db     24
761                                     <1> NumberOfBanks8:              db     15
762                                     <1> MemoryModel8:                db     VBE_MEMORYMODEL_DIRECT_COLOR
763                                     <1> BankSize8:            db     0
764                                     <1> NumberOfImagePages8:         db     17
765                                     <1> Reserved_page8:              db     0
766                                     <1> RedMaskSize8:                db     8
767                                     <1> RedFieldPosition8:    db     16
768                                     <1> GreenMaskSize8:              db     8
769                                     <1> GreenFieldPosition8:         db     8
770                                     <1> BlueMaskSize8:               db     8
771                                     <1> BlueFieldPosition8:          db     0
772                                     <1> RsvdMaskSize8:               db     0
773                                     <1> RsvdFieldPosition8:          db     0
774                                     <1> DirectColorModeInfo8:        db     0
775                                     <1> PhysBasePtr8:                dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
776                                     <1> OffScreenMemOffset8:         dd     0
777                                     <1> OffScreenMemSize8:    dw     0
778                                     <1> LinBytesPerScanLine8:        dw     1920
779                                     <1> BnkNumberOfPages8:           db     0
780                                     <1> LinNumberOfPages8:           db     0
781                                     <1> LinRedMaskSize8:      db     8
782                                     <1> LinRedFieldPosition8:        db     16
783                                     <1> LinGreenMaskSize8:    db     8
784                                     <1> LinGreenFieldPosition8:      db     8
785                                     <1> LinBlueMaskSize8:     db     8
786                                     <1> LinBlueFieldPosition8:       db     0
787                                     <1> LinRsvdMaskSize8:     db     0
788                                     <1> LinRsvdFieldPosition8:       db     0
789                                     <1> MaxPixelClock8:              dd     0
790                                     <1>
791                                     <1>                       dw     0114h ; 800x600x16
792                                     <1> ModeAttributes9:      dw     MODE_ATTRIBUTES
793                                     <1> WinAAttributes9:      db     WINA_ATTRIBUTES
794                                     <1> WinBAttributes9:      db     0
795                                     <1> WinGranularity9:      dw     VBE_DISPI_BANK_SIZE_KB
796                                     <1> WinSize9:             dw     VBE_DISPI_BANK_SIZE_KB
797                                     <1> WinASegment9:                dw     VGAMEM_GRAPH
798                                     <1> WinBSegment9:                dw     0000h
799                                     <1> WinFuncPtr9:          dd     0
800                                     <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
                        <1> NumberOfImagePages9:     db    16
                        <1> Reserved_page9:          db    0
                        <1> RedMaskSize9:            db    5
                        <1> RedFieldPosition9: db    11
                        <1> GreenMaskSize9:          db    6
                        <1> GreenFieldPosition9:     db    5
                        <1> BlueMaskSize9:           db    5
                        <1> BlueFieldPosition9:      db    0
                        <1> RsvdMaskSize9:           db    0
                        <1> RsvdFieldPosition9:      db    0
                        <1> DirectColorModeInfo9:    db    0
                        <1> PhysBasePtr9:            dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset9:     dd    0
                        <1> OffScreenMemSize9: dw    0
                        <1> LinBytesPerScanLine9:    dw    1600
                        <1> BnkNumberOfPages9:       db    0
                        <1> LinNumberOfPages9:       db    0
                        <1> LinRedMaskSize9:   db    5
                        <1> LinRedFieldPosition9:    db    11
                        <1> LinGreenMaskSize9: db    6
                        <1> LinGreenFieldPosition9:  db    5
                        <1> LinBlueMaskSize9:  db    5
                        <1> LinBlueFieldPosition9:   db    0
                        <1> LinRsvdMaskSize9:  db    0
                        <1> LinRsvdFieldPosition9:   db    0
                        <1> MaxPixelClock9:          dd    0
                        <1>
                        <1>                    dw    0115h ; 800x600x24
                        <1> ModeAttributes10:  dw    MODE_ATTRIBUTES
                        <1> WinAAttributes10:  db    WINA_ATTRIBUTES
                        <1> WinBAttributes10:  db    0
                        <1> WinGranularity10:  dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize10:         dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment10:           dw    VGAMEM_GRAPH
                        <1> WinBSegment10:           dw    0000h
                        <1> WinFuncPtr10:            dd    0
                        <1> BytesPerScanLine10:      dw    2400
                        <1> XResolution10:           dw    800
                        <1> YResolution10:           dw    600
                        <1> XCharSize10:       db    8
                        <1> YCharSize10:       db    16
                        <1> NumberOfPlanes10:  db    1
                        <1> BitsPerPixel10:          db    24
                        <1> NumberOfBanks10:   db    22
                        <1> MemoryModel10:           db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize10:        db    0
                        <1> NumberOfImagePages10:    db    10
                        <1> Reserved_page10:   db    0
                        <1> RedMaskSize10:           db    8
                        <1> RedFieldPosition10:      db    16
                        <1> GreenMaskSize10:   db    8
                        <1> GreenFieldPosition10:    db    8
                        <1> BlueMaskSize10:          db    8
                        <1> BlueFieldPosition10:     db    0
                        <1> RsvdMaskSize10:          db    0
                        <1> RsvdFieldPosition10:     db    0
                        <1> DirectColorModeInfo10:   db    0
                        <1> PhysBasePtr10:           dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
                        <1> OffScreenMemOffset10:    dd    0
                        <1> OffScreenMemSize10:      dw    0
                        <1> LinBytesPerScanLine10:   dw    2400
                        <1> BnkNumberOfPages10:      db    0
                        <1> LinNumberOfPages10:      db    0
                        <1> LinRedMaskSize10:  db    8
                        <1> LinRedFieldPosition10:   db    16
                        <1> LinGreenMaskSize10:      db    8
                        <1> LinGreenFieldPosition10:db       8
                        <1> LinBlueMaskSize10: db    8
                        <1> LinBlueFieldPosition10:  db    0
                        <1> LinRsvdMaskSize10: db    0
                        <1> LinRsvdFieldPosition10:  db    0
                        <1> MaxPixelClock10:   dd    0
                        <1>
                        <1>                    dw    0117h ; 1024x768x16
                        <1> ModeAttributes11:  dw    MODE_ATTRIBUTES
                        <1> WinAAttributes11:  db    WINA_ATTRIBUTES
                        <1> WinBAttributes11:  db    0
                        <1> WinGranularity11:  dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinSize11:         dw    VBE_DISPI_BANK_SIZE_KB
                        <1> WinASegment11:           dw    VGAMEM_GRAPH
                        <1> WinBSegment11:           dw    0000h
                        <1> WinFuncPtr11:            dd    0
                        <1> BytesPerScanLine11:      dw    2048
                        <1> XResolution11:           dw    1024
                        <1> YResolution11:           dw    768
                        <1> XCharSize11:       db    8
                        <1> YCharSize11:       db    16
                        <1> NumberOfPlanes11:  db    1
                        <1> BitsPerPixel11:          db    16
                        <1> NumberOfBanks11:   db    24
                        <1> MemoryModel11:           db    VBE_MEMORYMODEL_DIRECT_COLOR
                        <1> BankSize11:        db    0
                        <1> NumberOfImagePages11:    db    9
                        <1> Reserved_page11:   db    0
                        <1> RedMaskSize11:           db    5
                        <1> RedFieldPosition11:      db    11
```

```
                        <1> GreenMaskSize11:   db     6
                        <1> GreenFieldPosition11:     db     5
                        <1> BlueMaskSize11:           db     5
                        <1> BlueFieldPosition11:      db     0
                        <1> RsvdMaskSize11:           db     0
                        <1> RsvdFieldPosition11:      db     0
                        <1> DirectColorModeInfo11:    db     0
                        <1> PhysBasePtr11:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS,
                        <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
```

```
1011                            <1> LinRedMaskSize13:  db     8
1012                            <1> LinRedFieldPosition13:    db     16
1013                            <1> LinGreenMaskSize13:       db     8
1014                            <1> LinGreenFieldPosition13:db       8
1015                            <1> LinBlueMaskSize13: db     8
1016                            <1> LinBlueFieldPosition13:   db     0
1017                            <1> LinRsvdMaskSize13: db     8
1018                            <1> LinRsvdFieldPosition13:   db     24
1019                            <1> MaxPixelClock13:   dd     0
1020                            <1>
1021                            <1>                    dw     0141h ; 640x400x32
1022                            <1> ModeAttributes14:  dw     MODE_ATTRIBUTES
1023                            <1> WinAAttributes14:  db     WINA_ATTRIBUTES
1024                            <1> WinBAttributes14:  db     0
1025                            <1> WinGranularity14:  dw     VBE_DISPI_BANK_SIZE_KB
1026                            <1> WinSize14:         dw     VBE_DISPI_BANK_SIZE_KB
1027                            <1> WinASegment14:             dw     VGAMEM_GRAPH
1028                            <1> WinBSegment14:             dw     0000h
1029                            <1> WinFuncPtr14:              dd     0
1030                            <1> BytesPerScanLine14:        dw     2560
1031                            <1> XResolution14:             dw     640
1032                            <1> YResolution14:             dw     400
1033                            <1> XCharSize14:       db     8
1034                            <1> YCharSize14:       db     16
1035                            <1> NumberOfPlanes14:  db     1
1036                            <1> BitsPerPixel14:            db     32
1037                            <1> NumberOfBanks14:   db     16
1038                            <1> MemoryModel14:             db     VBE_MEMORYMODEL_DIRECT_COLOR
1039                            <1> BankSize14:        db     0
1040                            <1> NumberOfImagePages14:      db     15
1041                            <1> Reserved_page14:   db     0
1042                            <1> RedMaskSize14:             db     8
1043                            <1> RedFieldPosition14:        db     16
1044                            <1> GreenMaskSize14:   db     8
1045                            <1> GreenFieldPosition14:      db     8
1046                            <1> BlueMaskSize14:            db     8
1047                            <1> BlueFieldPosition14:       db     0
1048                            <1> RsvdMaskSize14:            db     8
1049                            <1> RsvdFieldPosition14:       db     24
1050                            <1> DirectColorModeInfo14:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
1051                            <1> PhysBasePtr14:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1052                            <1> OffScreenMemOffset14:      dd     0
1053                            <1> OffScreenMemSize14:        dw     0
1054                            <1> LinBytesPerScanLine14:     dw     2560
1055                            <1> BnkNumberOfPages14:        db     0
1056                            <1> LinNumberOfPages14:        db     0
1057                            <1> LinRedMaskSize14:  db     8
1058                            <1> LinRedFieldPosition14:     db     16
1059                            <1> LinGreenMaskSize14:        db     8
1060                            <1> LinGreenFieldPosition14:db       8
1061                            <1> LinBlueMaskSize14: db     8
1062                            <1> LinBlueFieldPosition14:    db     0
1063                            <1> LinRsvdMaskSize14: db     8
1064                            <1> LinRsvdFieldPosition14:    db     24
1065                            <1> MaxPixelClock14:   dd     0
1066                            <1>
1067                            <1>                    dw     0142 ; 640x480x32
1068                            <1> ModeAttributes15:  dw     MODE_ATTRIBUTES
1069                            <1> WinAAttributes15:  db     WINA_ATTRIBUTES
1070                            <1> WinBAttributes15:  db     0
1071                            <1> WinGranularity15:  dw     VBE_DISPI_BANK_SIZE_KB
1072                            <1> WinSize15:         dw     VBE_DISPI_BANK_SIZE_KB
1073                            <1> WinASegment15:             dw     VGAMEM_GRAPH
1074                            <1> WinBSegment15:             dw     0000h
1075                            <1> WinFuncPtr15:              dd     0
1076                            <1> BytesPerScanLine15:        dw     2560
1077                            <1> XResolution15:             dw     640
1078                            <1> YResolution15:             dw     480
1079                            <1> XCharSize15:       db     8
1080                            <1> YCharSize15:       db     16
1081                            <1> NumberOfPlanes15:  db     1
1082                            <1> BitsPerPixel15:            db     32
1083                            <1> NumberOfBanks15:   db     19
1084                            <1> MemoryModel15:             db     VBE_MEMORYMODEL_DIRECT_COLOR
1085                            <1> BankSize15:        db     0
1086                            <1> NumberOfImagePages15:      db     12
1087                            <1> Reserved_page15:   db     0
1088                            <1> RedMaskSize15:             db     8
1089                            <1> RedFieldPosition15:        db     16
1090                            <1> GreenMaskSize15:   db     8
1091                            <1> GreenFieldPosition15:      db     8
1092                            <1> BlueMaskSize15:            db     8
1093                            <1> BlueFieldPosition15:       db     0
1094                            <1> RsvdMaskSize15:            db     8
1095                            <1> RsvdFieldPosition15:       db     24
1096                            <1> DirectColorModeInfo15:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
1097                            <1> PhysBasePtr15:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS,
1098                            <1> OffScreenMemOffset15:      dd     0
1099                            <1> OffScreenMemSize15:        dw     0
1100                            <1> LinBytesPerScanLine15:     dw     2560
1101                            <1> BnkNumberOfPages15:        db     0
1102                            <1> LinNumberOfPages15:        db     0
1103                            <1> LinRedMaskSize15:  db     8
1104                            <1> LinRedFieldPosition15:     db     16
1105                            <1> LinGreenMaskSize15:        db     8
1106                            <1> LinGreenFieldPosition15:db       8
1107                            <1> LinBlueMaskSize15: db     8
1108                            <1> LinBlueFieldPosition15:    db     0
1109                            <1> LinRsvdMaskSize15: db     8
1110                            <1> LinRsvdFieldPosition15:    db     24
1111                            <1> MaxPixelClock15:   dd     0
1112                            <1>
1113                            <1>                    dw     0143h ; 800x600x32
1114                            <1> ModeAttributes16:  dw     MODE_ATTRIBUTES
1115                            <1> WinAAttributes16:  db     WINA_ATTRIBUTES
```

```
1116                            <1> WinBAttributes16:  db     0
1117                            <1> WinGranularity16:  dw     VBE_DISPI_BANK_SIZE_KB
1118                            <1> WinSize16:         dw     VBE_DISPI_BANK_SIZE_KB
1119                            <1> WinASegment16:             dw     VGAMEM_GRAPH
1120                            <1> WinBSegment16:             dw     0000h
1121                            <1> WinFuncPtr16:              dd     0
1122                            <1> BytesPerScanLine16:        dw     3200
1123                            <1> XResolution16:             dw     800
1124                            <1> YResolution16:             dw     600
1125                            <1> XCharSize16:       db     8
1126                            <1> YCharSize16:       db     16
1127                            <1> NumberOfPlanes16:  db     1
1128                            <1> BitsPerPixel16:            db     32
1129                            <1> NumberOfBanks16:   db     30
1130                            <1> MemoryModel16:             db     VBE_MEMORYMODEL_DIRECT_COLOR
1131                            <1> BankSize16:        db     0
1132                            <1> NumberOfImagePages16:      db     7
1133                            <1> Reserved_page16:   db     0
1134                            <1> RedMaskSize16:             db     8
1135                            <1> RedFieldPosition16:        db     16
1136                            <1> GreenMaskSize16:   db     8
1137                            <1> GreenFieldPosition16:      db     8
1138                            <1> BlueMaskSize16:            db     8
1139                            <1> BlueFieldPosition16:       db     0
1140                            <1> RsvdMaskSize16:            db     8
1141                            <1> RsvdFieldPosition16:       db     24
1142                            <1> DirectColorModeInfo16:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE,
1143                            <1> PhysBasePtr16:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS,
1144                            <1> OffScreenMemOffset16:      dd     0
1145                            <1> OffScreenMemSize16:        dw     0
1146                            <1> LinBytesPerScanLine16:     dw     3200
1147                            <1> BnkNumberOfPages16:        db     0
1148                            <1> LinNumberOfPages16:        db     0
1149                            <1> LinRedMaskSize16:  db     8
1150                            <1> LinRedFieldPosition16:     db     16
1151                            <1> LinGreenMaskSize16:        db     8
1152                            <1> LinGreenFieldPosition16:db       8
1153                            <1> LinBlueMaskSize16: db     8
1154                            <1> LinBlueFieldPosition16:    db     0
1155                            <1> LinRsvdMaskSize16: db     8
1156                            <1> LinRsvdFieldPosition16:    db     24
1157                            <1> MaxPixelClock16:   dd     0
1158                            <1>
1159                            <1>                    dw     0144h ; 1024x768x32
1160                            <1> ModeAttributes17:  dw     MODE_ATTRIBUTES
1161                            <1> WinAAttributes17:  db     WINA_ATTRIBUTES
1162                            <1> WinBAttributes17:  db     0
1163                            <1> WinGranularity17:  dw     VBE_DISPI_BANK_SIZE_KB
1164                            <1> WinSize17:         dw     VBE_DISPI_BANK_SIZE_KB
1165                            <1> WinASegment17:             dw     VGAMEM_GRAPH
1166                            <1> WinBSegment17:             dw     0000h
1167                            <1> WinFuncPtr17:              dd     0
1168                            <1> BytesPerScanLine17:        dw     4096
1169                            <1> XResolution17:             dw     1024
1170                            <1> YResolution17:             dw     768
1171                            <1> XCharSize17:       db     8
1172                            <1> YCharSize17:       db     16
1173                            <1> NumberOfPlanes17:  db     1
1174                            <1> BitsPerPixel17:            db     32
1175                            <1> NumberOfBanks17:   db     48
1176                            <1> MemoryModel17:             db     VBE_MEMORYMODEL_DIRECT_COLOR
1177                            <1> BankSize17:        db     0
1178                            <1> NumberOfImagePages17:      db     4
1179                            <1> Reserved_page17:   db     0
1180                            <1> RedMaskSize17:             db     8
1181                            <1> RedFieldPosition17:        db     16
1182                            <1> GreenMaskSize17:   db     8
1183                            <1> GreenFieldPosition17:      db     8
1184                            <1> BlueMaskSize17:            db     8
1185                            <1> BlueFieldPosition17:       db     0
1186                            <1> RsvdMaskSize17:            db     8
1187                            <1> RsvdFieldPosition17:       db     24
1188                            <1> DirectColorModeInfo17:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
1189                            <1> PhysBasePtr17:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1190                            <1> OffScreenMemOffset17:      dd     0
1191                            <1> OffScreenMemSize17:        dw     0
1192                            <1> LinBytesPerScanLine17:     dw     4096
1193                            <1> BnkNumberOfPages17:        db     0
1194                            <1> LinNumberOfPages17:        db     0
1195                            <1> LinRedMaskSize17:  db     8
1196                            <1> LinRedFieldPosition17:     db     16
1197                            <1> LinGreenMaskSize17:        db     8
1198                            <1> LinGreenFieldPosition17:db       8
1199                            <1> LinBlueMaskSize17: db     8
1200                            <1> LinBlueFieldPosition17:    db     0
1201                            <1> LinRsvdMaskSize17: db     8
1202                            <1> LinRsvdFieldPosition17:    db     24
1203                            <1> MaxPixelClock17:   dd     0
1204                            <1>
1205                            <1>                    dw     0146h ; 320x200x8
1206                            <1> ModeAttributes18:  dw     MODE_ATTRIBUTES
1207                            <1> WinAAttributes18:  db     WINA_ATTRIBUTES
1208                            <1> WinBAttributes18:  db     0
1209                            <1> WinGranularity18:  dw     VBE_DISPI_BANK_SIZE_KB
1210                            <1> WinSize18:         dw     VBE_DISPI_BANK_SIZE_KB
1211                            <1> WinASegment18:             dw     VGAMEM_GRAPH
1212                            <1> WinBSegment18:             dw     0000h
1213                            <1> WinFuncPtr18:              dd     0
1214                            <1> BytesPerScanLine18:        dw     320
1215                            <1> XResolution18:             dw     320
1216                            <1> YResolution18:             dw     200
1217                            <1> XCharSize18:       db     8
1218                            <1> YCharSize18:       db     16
1219                            <1> NumberOfPlanes18:  db     1
1220                            <1> BitsPerPixel18:            db     8
```

```
1221                            <1> NumberOfBanks18:    db     1
1222                            <1> MemoryModel18:              db     VBE_MEMORYMODEL_PACKED_PIXEL
1223                            <1> BankSize18:         db     0
1224                            <1> NumberOfImagePages18:       db     255 ; 261 in vbetables.h (03/01/2020) !
1225                            <1> Reserved_page18:    db     0
1226                            <1> RedMaskSize18:              db     0
1227                            <1> RedFieldPosition18:         db     0
1228                            <1> GreenMaskSize18:    db     0
1229                            <1> GreenFieldPosition18:       db     0
1230                            <1> BlueMaskSize18:             db     0
1231                            <1> BlueFieldPosition18:        db     0
1232                            <1> RsvdMaskSize18:             db     0
1233                            <1> RsvdFieldPosition18:        db     0
1234                            <1> DirectColorModeInfo18:      db     0
1235                            <1> PhysBasePtr18:              dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1236                            <1> OffScreenMemOffset18:       dd     0
1237                            <1> OffScreenMemSize18:         dw     0
1238                            <1> LinBytesPerScanLine18:      dw     320
1239                            <1> BnkNumberOfPages18:         db     0
1240                            <1> LinNumberOfPages18:         db     0
1241                            <1> LinRedMaskSize18:   db     0
1242                            <1> LinRedFieldPosition18:      db     0
1243                            <1> LinGreenMaskSize18:         db     0
1244                            <1> LinGreenFieldPosition18:db         0
1245                            <1> LinBlueMaskSize18:          db     0
1246                            <1> LinBlueFieldPosition18: db         0
1247                            <1> LinRsvdMaskSize18:          db     0
1248                            <1> LinRsvdFieldPosition18: db         0
1249                            <1> MaxPixelClock18:    dd     0
1250                            <1>
1251                            <1>                     dw     018Dh ; 1280x720x16
1252                            <1> ModeAttributes19:   dw     MODE_ATTRIBUTES
1253                            <1> WinAAttributes19:   db     WINA_ATTRIBUTES
1254                            <1> WinBAttributes19:   db     0
1255                            <1> WinGranularity19:   dw     VBE_DISPI_BANK_SIZE_KB
1256                            <1> WinSize19:          dw     VBE_DISPI_BANK_SIZE_KB
1257                            <1> WinASegment19:              dw     VGAMEM_GRAPH
1258                            <1> WinBSegment19:              dw     0000h
1259                            <1> WinFuncPtr19:               dd     0
1260                            <1> BytesPerScanLine19:         dw     2560
1261                            <1> XResolution19:              dw     1280
1262                            <1> YResolution19:              dw     720
1263                            <1> XCharSize19:        db     8
1264                            <1> YCharSize19:        db     16
1265                            <1> NumberOfPlanes19:   db     1
1266                            <1> BitsPerPixel19:             db     16
1267                            <1> NumberOfBanks19:    db     29
1268                            <1> MemoryModel19:              db     VBE_MEMORYMODEL_DIRECT_COLOR
1269                            <1> BankSize19:         db     0
1270                            <1> NumberOfImagePages19:       db     8
1271                            <1> Reserved_page19:    db     0
1272                            <1> RedMaskSize19:              db     5
1273                            <1> RedFieldPosition19:         db     11
1274                            <1> GreenMaskSize19:    db     6
1275                            <1> GreenFieldPosition19:       db     5
1276                            <1> BlueMaskSize19:             db     5
1277                            <1> BlueFieldPosition19:        db     0
1278                            <1> RsvdMaskSize19:             db     0
1279                            <1> RsvdFieldPosition19:        db     0
1280                            <1> DirectColorModeInfo19:      db     0
1281                            <1> PhysBasePtr19:              dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1282                            <1> OffScreenMemOffset19:       dd     0
1283                            <1> OffScreenMemSize19:         dw     0
1284                            <1> LinBytesPerScanLine19:      dw     2560
1285                            <1> BnkNumberOfPages19:         db     0
1286                            <1> LinNumberOfPages19:         db     0
1287                            <1> LinRedMaskSize19:   db     5
1288                            <1> LinRedFieldPosition19:      db     11
1289                            <1> LinGreenMaskSize19:         db     6
1290                            <1> LinGreenFieldPosition19:db         5
1291                            <1> LinBlueMaskSize19:  db     5
1292                            <1> LinBlueFieldPosition19:     db     0
1293                            <1> LinRsvdMaskSize19:  db     0
1294                            <1> LinRsvdFieldPosition19:     db     0
1295                            <1> MaxPixelClock19:    dd     0
1296                            <1>
1297                            <1>                     dw     018Eh ; 1280x720x24
1298                            <1> ModeAttributes20:   dw     MODE_ATTRIBUTES
1299                            <1> WinAAttributes20:   db     WINA_ATTRIBUTES
1300                            <1> WinBAttributes20:   db     0
1301                            <1> WinGranularity20:   dw     VBE_DISPI_BANK_SIZE_KB
1302                            <1> WinSize20:          dw     VBE_DISPI_BANK_SIZE_KB
1303                            <1> WinASegment20:              dw     VGAMEM_GRAPH
1304                            <1> WinBSegment20:              dw     0000h
1305                            <1> WinFuncPtr20:               dd     0
1306                            <1> BytesPerScanLine20:         dw     3840
1307                            <1> XResolution20:              dw     1280
1308                            <1> YResolution20:              dw     720
1309                            <1> XCharSize20:        db     8
1310                            <1> YCharSize20:        db     16
1311                            <1> NumberOfPlanes20:   db     1
1312                            <1> BitsPerPixel20:             db     24
1313                            <1> NumberOfBanks20:    db     43
1314                            <1> MemoryModel20:              db     VBE_MEMORYMODEL_DIRECT_COLOR
1315                            <1> BankSize20:         db     0
1316                            <1> NumberOfImagePages20:       db     5
1317                            <1> Reserved_page20:    db     0
1318                            <1> RedMaskSize20:              db     8
1319                            <1> RedFieldPosition20:         db     16
1320                            <1> GreenMaskSize20:    db     8
1321                            <1> GreenFieldPosition20:       db     8
1322                            <1> BlueMaskSize20:             db     8
1323                            <1> BlueFieldPosition20:        db     0
1324                            <1> RsvdMaskSize20:             db     0
1325                            <1> RsvdFieldPosition20:        db     0
```

```
1326                            <1> DirectColorModeInfo20:    db     0
1327                            <1> PhysBasePtr20:            dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1328                            <1> OffScreenMemOffset20:     dd     0
1329                            <1> OffScreenMemSize20:       dw     0
1330                            <1> LinBytesPerScanLine20:    dw     3840
1331                            <1> BnkNumberOfPages20:       db     0
1332                            <1> LinNumberOfPages20:       db     0
1333                            <1> LinRedMaskSize20:  db     8
1334                            <1> LinRedFieldPosition20:    db     16
1335                            <1> LinGreenMaskSize20:       db     8
1336                            <1> LinGreenFieldPosition20:db       8
1337                            <1> LinBlueMaskSize20: db     8
1338                            <1> LinBlueFieldPosition20:   db     0
1339                            <1> LinRsvdMaskSize20: db     0
1340                            <1> LinRsvdFieldPosition20:   db     0
1341                            <1> MaxPixelClock20:   dd     0
1342                            <1>
1343                            <1>                    dw     018Fh ; 1280x720x32
1344                            <1> ModeAttributes21:  dw     MODE_ATTRIBUTES
1345                            <1> WinAAttributes21:  db     WINA_ATTRIBUTES
1346                            <1> WinBAttributes21:  db     0
1347                            <1> WinGranularity21:  dw     VBE_DISPI_BANK_SIZE_KB
1348                            <1> WinSize21:         dw     VBE_DISPI_BANK_SIZE_KB
1349                            <1> WinASegment21:             dw     VGAMEM_GRAPH
1350                            <1> WinBSegment21:             dw     0000h
1351                            <1> WinFuncPtr21:              dd     0
1352                            <1> BytesPerScanLine21:        dw     5120
1353                            <1> XResolution21:             dw     1280
1354                            <1> YResolution21:             dw     720
1355                            <1> XCharSize21:       db     8
1356                            <1> YCharSize21:       db     16
1357                            <1> NumberOfPlanes21:  db     1
1358                            <1> BitsPerPixel21:            db     32
1359                            <1> NumberOfBanks21:   db     57
1360                            <1> MemoryModel21:             db     VBE_MEMORYMODEL_DIRECT_COLOR
1361                            <1> BankSize21:        db     0
1362                            <1> NumberOfImagePages21:      db     3
1363                            <1> Reserved_page21:   db     0
1364                            <1> RedMaskSize21:             db     8
1365                            <1> RedFieldPosition21:        db     16
1366                            <1> GreenMaskSize21:   db     8
1367                            <1> GreenFieldPosition21:      db     8
1368                            <1> BlueMaskSize21:            db     8
1369                            <1> BlueFieldPosition21:       db     0
1370                            <1> RsvdMaskSize21:            db     8
1371                            <1> RsvdFieldPosition21:       db     24
1372                            <1> DirectColorModeInfo21:     db     VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
1373                            <1> PhysBasePtr21:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1374                            <1> OffScreenMemOffset21:      dd     0
1375                            <1> OffScreenMemSize21:        dw     0
1376                            <1> LinBytesPerScanLine21:     dw     5120
1377                            <1> BnkNumberOfPages21:        db     0
1378                            <1> LinNumberOfPages21:        db     0
1379                            <1> LinRedMaskSize21:  db     8
1380                            <1> LinRedFieldPosition21:     db     16
1381                            <1> LinGreenMaskSize21:        db     8
1382                            <1> LinGreenFieldPosition21:db       8
1383                            <1> LinBlueMaskSize21: db     8
1384                            <1> LinBlueFieldPosition21:    db     0
1385                            <1> LinRsvdMaskSize21: db     8
1386                            <1> LinRsvdFieldPosition21:    db     24
1387                            <1> MaxPixelClock21:   dd     0
1388                            <1>
1389                            <1>                    dw     0190h ; 1920x1080x16
1390                            <1> ModeAttributes22:  dw     MODE_ATTRIBUTES
1391                            <1> WinAAttributes22:  db     WINA_ATTRIBUTES
1392                            <1> WinBAttributes22:  db     0
1393                            <1> WinGranularity22:  dw     VBE_DISPI_BANK_SIZE_KB
1394                            <1> WinSize22:         dw     VBE_DISPI_BANK_SIZE_KB
1395                            <1> WinASegment22:             dw     VGAMEM_GRAPH
1396                            <1> WinBSegment22:             dw     0000h
1397                            <1> WinFuncPtr22:              dd     0
1398                            <1> BytesPerScanLine22:        dw     3840
1399                            <1> XResolution22:             dw     1920
1400                            <1> YResolution22:             dw     1080
1401                            <1> XCharSize22:       db     8
1402                            <1> YCharSize22:       db     16
1403                            <1> NumberOfPlanes22:  db     1
1404                            <1> BitsPerPixel22:            db     16
1405                            <1> NumberOfBanks22:   db     64
1406                            <1> MemoryModel22:             db     VBE_MEMORYMODEL_DIRECT_COLOR,
1407                            <1> BankSize22:        db     0
1408                            <1> NumberOfImagePages22:      db     3
1409                            <1> Reserved_page22:   db     0
1410                            <1> RedMaskSize22:             db     5
1411                            <1> RedFieldPosition22:        db     11
1412                            <1> GreenMaskSize22:   db     6
1413                            <1> GreenFieldPosition22:      db     5
1414                            <1> BlueMaskSize22:            db     5
1415                            <1> BlueFieldPosition22:       db     0
1416                            <1> RsvdMaskSize22:            db     0
1417                            <1> RsvdFieldPosition22:       db     0
1418                            <1> DirectColorModeInfo22:     db     0
1419                            <1> PhysBasePtr22:             dd     VBE_DISPI_LFB_PHYSICAL_ADDRESS
1420                            <1> OffScreenMemOffset22:      dd     0
1421                            <1> OffScreenMemSize22:        dw     0
1422                            <1> LinBytesPerScanLine22:     dw     3840
1423                            <1> BnkNumberOfPages22:        db     0
1424                            <1> LinNumberOfPages22:        db     0
1425                            <1> LinRedMaskSize22:  db     5
1426                            <1> LinRedFieldPosition22:     db     11
1427                            <1> LinGreenMaskSize22:        db     6
1428                            <1> LinGreenFieldPosition22:db       5
1429                            <1> LinBlueMaskSize22: db     5
1430                            <1> LinBlueFieldPosition22:    db     0
```

```
1431                            <1> LinRsvdMaskSize22: db    0
1432                            <1> LinRsvdFieldPosition22:  db    0
1433                            <1> MaxPixelClock22:   dd    0
1434                            <1>
1435                            <1>                    dw    0191h ; 1920x1080x24
1436                            <1> ModeAttributes23:  dw    MODE_ATTRIBUTES
1437                            <1> WinAAttributes23:  db    WINA_ATTRIBUTES
1438                            <1> WinBAttributes23:  db    0
1439                            <1> WinGranularity23:  dw    VBE_DISPI_BANK_SIZE_KB
1440                            <1> WinSize23:         dw    VBE_DISPI_BANK_SIZE_KB
1441                            <1> WinASegment23:             dw    VGAMEM_GRAPH
1442                            <1> WinBSegment23:             dw    0000h
1443                            <1> WinFuncPtr23:              dd    0
1444                            <1> BytesPerScanLine23:        dw    5760
1445                            <1> XResolution23:             dw    1920
1446                            <1> YResolution23:             dw    1080
1447                            <1> XCharSize23:       db    8
1448                            <1> YCharSize23:       db    16
1449                            <1> NumberOfPlanes23:  db    1
1450                            <1> BitsPerPixel23:            db    24
1451                            <1> NumberOfBanks23:   db    95
1452                            <1> MemoryModel23:             db    VBE_MEMORYMODEL_DIRECT_COLOR
1453                            <1> BankSize23:        db    0
1454                            <1> NumberOfImagePages23:      db    1
1455                            <1> Reserved_page23:   db    0
1456                            <1> RedMaskSize23:             db    8
1457                            <1> RedFieldPosition23:        db    16
1458                            <1> GreenMaskSize23:   db    8
1459                            <1> GreenFieldPosition23:      db    8
1460                            <1> BlueMaskSize23:            db    8
1461                            <1> BlueFieldPosition23:       db    0
1462                            <1> RsvdMaskSize23:            db    0
1463                            <1> RsvdFieldPosition23:       db    0
1464                            <1> DirectColorModeInfo23:     db    0
1465                            <1> PhysBasePtr23:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
1466                            <1> OffScreenMemOffset23:      dd    0
1467                            <1> OffScreenMemSize23:        dw    0
1468                            <1> LinBytesPerScanLine23:     dw    5760
1469                            <1> BnkNumberOfPages23:        db    0
1470                            <1> LinNumberOfPages23:        db    0
1471                            <1> LinRedMaskSize23:  db    8
1472                            <1> LinRedFieldPosition23:     db    16
1473                            <1> LinGreenMaskSize23:        db    8
1474                            <1> LinGreenFieldPosition23:db       8
1475                            <1> LinBlueMaskSize23: db    8
1476                            <1> LinBlueFieldPosition23:    db    0
1477                            <1> LinRsvdMaskSize23: db    0
1478                            <1> LinRsvdFieldPosition23:    db    0
1479                            <1> MaxPixelClock23:   dd    0
1480                            <1>
1481                            <1>                    dw    0192h ; 1920x1080x32
1482                            <1> ModeAttributes24:  dw    MODE_ATTRIBUTES
1483                            <1> WinAAttributes24:  db    WINA_ATTRIBUTES
1484                            <1> WinBAttributes24:  db    0
1485                            <1> WinGranularity24:  dw    VBE_DISPI_BANK_SIZE_KB
1486                            <1> WinSize24:         dw    VBE_DISPI_BANK_SIZE_KB
1487                            <1> WinASegment24:             dw    VGAMEM_GRAPH
1488                            <1> WinBSegment24:             dw    0000h
1489                            <1> WinFuncPtr24:              dd    0
1490                            <1> BytesPerScanLine24:        dw    7680
1491                            <1> XResolution24:             dw    1920
1492                            <1> YResolution24:             dw    1080
1493                            <1> XCharSize24:       db    8
1494                            <1> YCharSize24:       db    16
1495                            <1> NumberOfPlanes24:  db    1
1496                            <1> BitsPerPixel24:            db    32
1497                            <1> NumberOfBanks24:   db    127
1498                            <1> MemoryModel24:             db    VBE_MEMORYMODEL_DIRECT_COLOR
1499                            <1> BankSize24:        db    0
1500                            <1> NumberOfImagePages24:      db    1
1501                            <1> Reserved_page24:   db    0
1502                            <1> RedMaskSize24:             db    8
1503                            <1> RedFieldPosition24:        db    16
1504                            <1> GreenMaskSize24:   db    8
1505                            <1> GreenFieldPosition24:      db    8
1506                            <1> BlueMaskSize24:            db    8
1507                            <1> BlueFieldPosition24:       db    0
1508                            <1> RsvdMaskSize24:            db    8
1509                            <1> RsvdFieldPosition24:       db    24
1510                            <1> DirectColorModeInfo24:     db    VBE_DIRECTCOLOR_RESERVED_BITS_AVAILABLE
1511                            <1> PhysBasePtr24:             dd    VBE_DISPI_LFB_PHYSICAL_ADDRESS
1512                            <1> OffScreenMemOffset24:      dd    0
1513                            <1> OffScreenMemSize24:        dw    0
1514                            <1> LinBytesPerScanLine24:     dw    7680
1515                            <1> BnkNumberOfPages24:        db    0
1516                            <1> LinNumberOfPages24:        db    0
1517                            <1> LinRedMaskSize24:  db    8
1518                            <1> LinRedFieldPosition24:     db    16
1519                            <1> LinGreenMaskSize24:        db    8
1520                            <1> LinGreenFieldPosition24:db       8
1521                            <1> LinBlueMaskSize24: db    8
1522                            <1> LinBlueFieldPosition24:    db    0
1523                            <1> LinRsvdMaskSize24: db    8
1524                            <1> LinRsvdFieldPosition24:    db    24
1525                            <1> MaxPixelClock24:   dd    0
1526                            <1>
1527                            <1> VBE_VESA_MODE_END_OF_LIST: dw    0
1528                            <1>
1529                            <1> %endif
1530                            <1>
1531                            <1> ; 24/11/2020
1532                            <1>
1533                            <1> direct_color_fields:
1534                            <1>       ; 24/11/2020
1535                            <1>
```

```
1536                            <1> ; (vbetables-gen.c)
1537                            <1> ; // Direct Color fields
1538                            <1> ; (required for direct/6 and YUV/7 memory models)
1539                            <1> ; switch(pm->depth) {
1540                            <1>
1541                            <1> ;case 8:
1542 00007426 00                 <1> r_size_8:  db 0
1543 00007427 00                 <1> r_pos_8:   db 0
1544 00007428 00                 <1> g_size_8:  db 0
1545 00007429 00                 <1> g_pos_8:   db 0
1546 0000742A 00                 <1> b_size_8:  db 0
1547 0000742B 00                 <1> b_pos_8:   db 0
1548 0000742C 00                 <1> a_size_8:  db 0
1549 0000742D 00                 <1> a_pos_8:   db 0
1550                            <1>
1551                            <1> ;case 16:
1552 0000742E 05                 <1> r_size_16:  db 5
1553 0000742F 0B                 <1> r_pos_16:   db 11
1554 00007430 06                 <1> g_size_16:  db 6
1555 00007431 05                 <1> g_pos_16:   db 5
1556 00007432 05                 <1> b_size_16:  db 5
1557 00007433 00                 <1> b_pos_16:   db 0
1558 00007434 00                 <1> a_size_16:  db 0
1559 00007435 00                 <1> a_pos_16:   db 0
1560                            <1>
1561                            <1> ;case 24:
1562 00007436 08                 <1> r_size_24:  db 8
1563 00007437 10                 <1> r_pos_24:   db 16
1564 00007438 08                 <1> g_size_24:  db 8
1565 00007439 08                 <1> g_pos_24:   db 8
1566 0000743A 08                 <1> b_size_24:  db 8
1567 0000743B 00                 <1> b_pos_24:   db 0
1568 0000743C 00                 <1> a_size_24:  db 0
1569 0000743D 00                 <1> a_pos_24:   db 0
1570                            <1>
1571                            <1> ;case 32:
1572 0000743E 08                 <1> r_size_32:  db 8
1573 0000743F 10                 <1> r_pos_32:   db 16
1574 00007440 08                 <1> g_size_32:  db 8
1575 00007441 08                 <1> g_pos_32:   db 8
1576 00007442 08                 <1> b_size_32:  db 8
1577 00007443 00                 <1> b_pos_32:   db 0
1578 00007444 08                 <1> a_size_32:  db 8
1579 00007445 18                 <1> a_pos_32:   db 24
3080                                ;%include 'diskdata.s'    ; DISK (BIOS) DATA (initialized)
3081                                ;;;
3082
3083                                Align 2
3084
3085                                %include 'sysdefs.s' ; 24/01/2015
   1                            <1> ; ****************************************************************************
   2                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - SYSTEM DEFINITIONS : sysdefs.s
   3                            <1> ; ----------------------------------------------------------------------------
   4                            <1> ; Last Update: 31/12/2017
   5                            <1> ; ----------------------------------------------------------------------------
   6                            <1> ; Beginning: 24/01/2016
   7                            <1> ; ----------------------------------------------------------------------------
   8                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                            <1> ; ----------------------------------------------------------------------------
  10                            <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  11                            <1> ; sysdefs.inc (14/11/2015)
  12                            <1> ; ****************************************************************************
  13                            <1>
  14                            <1> ; Retro UNIX 386 v1 Kernel - SYSDEFS.INC
  15                            <1> ; Last Modification: 14/11/2015
  16                            <1> ;
  17                            <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
  18                            <1> ; (Modified from
  19                            <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
  20                            <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
  21                            <1> ;     UNIX.ASM (MASM 6.11) --> SYSDEFS.INC (NASM 2.11)
  22                            <1> ; ----------------------------------------------------------------------------
  23                            <1> ;
  24                            <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
  25                            <1> ; (Original) Source Code by Ken Thompson (1971-1972)
  26                            <1> ; <Bell Laboratories (17/3/1972)>
  27                            <1> ; <Preliminary Release of UNIX Implementation Document>
  28                            <1> ;
  29                            <1> ; ****************************************************************************
  30                            <1>
  31                            <1> nproc       equ    16  ; number of processes
  32                            <1> nfiles      equ    50
  33                            <1> ntty  equ     8   ; 8+1 -> 8 (10/05/2013)
  34                            <1> nbuf  equ    4   ; 6 ;; 21/08/2015 - 'namei' buffer problem when nbuf > 4
  35                            <1>             ; NOTE: If fd0 super block buffer addres is beyond of the 1st
  36                            <1>             ; 32K, DMA r/w routine or someting else causes a jump to
  37                            <1>             ; kernel panic routine (in 'alloc' routine, in u5.s)
  38                            <1>             ; because of invalid buffer content (r/w error).
  39                            <1>             ; When all buffers are set before the end of the 1st 32k,
  40                            <1>             ; there is no problem!? (14/11/2015)
  41                            <1>
  42                            <1> ;csgmnt     equ    2000h  ; 26/05/2013 (segment of process 1)
  43                            <1> ;core equ    0          ; 19/04/2013
  44                            <1> ;ecore      equ    32768 - 64  ; 04/06/2013 (24/05/2013)
  45                            <1>       ; (if total size of argument list and arguments is 128 bytes)
  46                            <1>       ; maximum executable file size = 32768-(64+40+128-6) = 32530 bytes
  47                            <1>       ; maximum stack size = 40 bytes (+6 bytes for 'IRET' at 32570)
  48                            <1>       ; initial value of user's stack pointer = 32768-64-128-2 = 32574
  49                            <1>       ;     (sp=32768-args_space-2 at the beginning of execution)
  50                            <1>       ; argument list offset = 32768-64-128 = 32576 (if it is 128 bytes)
  51                            <1>       ; 'u' structure offset (for the '/core' dump file) = 32704
  52                            <1>       ; '/core' dump file size = 32768 bytes
  53                            <1>
  54                            <1> ; 08/03/2014
  55                            <1> ;sdsegmnt equ      6C0h  ; 256*16 bytes (swap data segment size for 16 processes)
```

```
<1> ; 19/04/2013 Retro UNIX 8086 v1 feaure only !
<1> ;;sdsegmnt equ     740h  ; swap data segment (for user structures and registers)
<1>
<1> ; 30/08/2013
<1> time_count equ 4 ; 10 --> 4 01/02/2014
<1>
<1> ; 05/02/2014
<1> ; process status
<1> ;SFREE      equ 0
<1> ;SRUN equ 1
<1> ;SWAIT      equ 2
<1> ;SZOMB      equ 3
<1> ;SSLEEP     equ 4 ; Retro UNIX 8086 V1 extension (for sleep and wakeup)
<1>
<1> ; 09/03/2015
<1> userdata equ 80000h ; user structure data address for current user ; temporary
<1> swap_queue equ 90000h - 2000h ; swap queue address ; temporary
<1> swap_alloc_table equ 0D0000h  ;  swap allocation table address ; temporary
<1>
<1> ; 17/09/2015
<1> ESPACE equ 48 ; [u.usp] (at 'sysent') - [u.sp] value for error return
<1>
<1> ; 31/12/2017
<1> ; 19/02/2017
<1> ; 15/10/2016
<1> ; 20/05/2016
<1> ; 19/05/2016
<1> ; 18/05/2016
<1> ; 29/04/2016
<1> ; TRDOS 386 (TRDOS v2.0) system calls - temporary List
<1> ; 14/07/2013 - 21/09/2015 (Retro UNIX 8086 & 386 system calls)
<1> _ver  equ 0 ; Get TRDOS version (v2.0)
<1> _exit       equ 1
<1> _fork       equ 2
<1> _read       equ 3
<1> _write      equ 4
<1> _open equ 5
<1> _close      equ 6
<1> _wait       equ 7
<1> _creat      equ 8
<1> _rename     equ 9  ; TRDOS 386, Rename File (31/12/2017)
<1> _delete     equ 10 ; TRDOS 386, Delete File (29/12/2017)
<1> _exec equ 11
<1> _chdir      equ 12
<1> _time       equ 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
<1> _mkdir      equ 14
<1> _chmod      equ 15 ; TRDOS 386, Change Attributes (30/12/2017)
<1> _rmdir      equ 16 ; TRDOS 386, Remove Directory (29/12/2017)
<1> _break      equ 17
<1> _drive      equ 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
<1> _seek equ 19
<1> _tell       equ 20
<1> _mem  equ 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
<1> _prompt     equ 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
<1> _path equ 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
<1> _env  equ 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
<1> _stime      equ 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
<1> _quit equ 26
<1> _intr equ 27
<1> _dir  equ 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
<1> _emt  equ 29
<1> _ldrvt      equ 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
<1> _video  equ 31 ; TRDOS 386 Video Functions (16/05/2016)
<1> _audio      equ 32 ; TRDOS 386 Video Functions (16/05/2016)
<1> _timer      equ 33 ; TRDOS 386 Timer Functions (18/05/2016)
<1> _sleep      equ 34 ; Retro UNIX 8086 v1 feature only !
<1> _msg  equ 35 ; Retro UNIX 386 v1 feature only !
<1> _geterr     equ 36 ; Retro UNIX 386 v1 feature only !
<1> _fpsave equ 37 ; TRDOS 386 FPU state option (28/02/2017)
<1> _pri  equ 38 ; change priority - TRDOS 386 (20/05/2016)
<1> _rele equ 39 ; TRDOS 386 (19/05/2016)
<1> _fff  equ 40 ; Find First File - TRDOS 386 (15/10/2016)
<1> _fnf  equ 41 ; Find Next File - TRDOS 386 (15/10/2016)
<1> _alloc      equ 42 ; Allocate memory - TRDOS 386 (19/02/2017)
<1>             ; TRDOS 386 (19/02/2017) DMA buff fuctions
<1> _dalloc equ 43 ; Deallocate mem - TRDOS 386 (19/02/2017)
<1> _calbac equ 44 ; Set IRQ callback - TRDOS 386 (20/02/2017)
<1> _dma  equ 45 ; DMA service - TRDOS 386 (20/08/2017)
<1>
<1> %macro sys 1-4
<1>     ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
<1>     ; 03/09/2015
<1>     ; 13/04/2015
<1>     ; Retro UNIX 386 v1 system call.
<1>     %if %0 >= 2
<1>         mov ebx, %2
<1>         %if %0 >= 3
<1>             mov ecx, %3
<1>             %if %0 = 4
<1>                mov edx, %4
<1>             %endif
<1>         %endif
<1>     %endif
<1>     mov eax, %1
<1>     ;int 30h
<1>     int 40h ; TRDOS 386 (TRDOS v2.0)
<1> %endmacro
<1>
<1> ; TRDOS 386 system calls, interrupt number
<1> ; 25/12/2016
<1> SYSCALL_INT_NUM   equ '40' ; '40h'
<1>
<1> ; 13/05/2015 - ERROR CODES
<1> ERR_FILE_NOT_OPEN  equ 10 ; 'file not open !' error
<1> ERR_FILE_ACCESS    equ 11 ; 'permission denied !' error
```

```
                        <1> ; 14/05/2015
                        <1> ERR_DIR_ACCESS     equ 11 ; 'permission denied !' error
                        <1> ERR_FILE_NOT_FOUND equ 12 ; 'file not found !' error
                        <1> ERR_TOO_MANY_FILES equ 13 ; 'too many open files !' error
                        <1> ERR_DIR_EXISTS     equ 14 ; 'directory already exists !' error
                        <1> ; 16/05/2015
                        <1> ERR_DRV_NOT_RDY    equ 15 ; 'drive not ready !' error
                        <1> ; 18/05/2015
                        <1> ERR_DEV_NOT_RDY    equ 15 ; 'device not ready !' error
                        <1> ERR_DEV_ACCESS     equ 11 ; 'permission denied !' error
                        <1> ERR_DEV_NOT_OPEN   equ 10 ; 'device not open !' error
                        <1> ; 07/06/2015
                        <1> ERR_FILE_EOF    equ 16 ; 'end of file !' error
                        <1> ERR_DEV_VOL_SIZE   equ 16 ; 'out of volume !' error
                        <1> ; 09/06/2015
                        <1> ERR_DRV_READ    equ 17 ; 'disk read error !'
                        <1> ERR_DRV_WRITE         equ 18 ; 'disk write error !'
                        <1> ; 16/06/2015
                        <1> ERR_NOT_DIR     equ 19 ; 'not a (valid) directory !' error
                        <1> ERR_FILE_SIZE         equ 20 ; 'file size error !'
                        <1> ; 22/06/2015
                        <1> ERR_NOT_SUPERUSER  equ 11 ; 'permission denied !' error
                        <1> ERR_NOT_OWNER      equ 11 ; 'permission denied !' error
                        <1> ERR_NOT_FILE       equ 11 ; 'permission denied !' error
                        <1> ; 23/06/2015
                        <1> ERR_FILE_EXISTS    equ 14 ; 'file already exists !' error
                        <1> ERR_DRV_NOT_SAME   equ 21 ; 'not same drive !' error
                        <1> ERR_DIR_NOT_FOUND  equ 12 ; 'directory not found !' error
                        <1> ERR_NOT_EXECUTABLE equ 22 ; 'not executable file !' error
                        <1> ; 27/06/2015
                        <1> ERR_INV_PARAMETER  equ 23 ; 'invalid parameter !' error
                        <1> ERR_INV_DEV_NAME   equ 24 ; 'invalid device name !' error
                        <1> ; 29/06/2015
                        <1> ERR_TIME_OUT    equ 25 ; 'time out !' error
                        <1> ERR_DEV_NOT_RESP   equ 25 ; 'device not responding !' error
                        <1> ; 10/10/2016
                        <1> ERR_INV_FILE_NAME  equ 26 ; 'invalid file name !' error
                        <1> ERR_INV_FLAGS         equ 23 ; 'invalid flags !' error
                        <1> ; For code compatibility with previous version of TRDOS (2011)
                        <1> ; (Temporary error codes for current TRDOS 386 -2016- version)
                        <1> ERR_NO_MORE_FILES  equ 12 ; 'no more files !' error
                        <1> ERR_PATH_NOT_FOUND equ  3 ; 'path not found !' error
                        <1>                         ; 'dir not found !' ; TRDOS 8086
                        <1> ERR_NOT_FOUND:         equ  2 ; 'file not found !' ; TRDOS 8086
                        <1> ERR_DISK_SPACE         equ 39 ; 'out of volume !' TRDOS 8086
                        <1>                        ; 'insufficient disk space !' ; 27h
                        <1> ERR_DISK_WRITE         equ 30 ; 'disk write protected !' ; 16/10/2016
                        <1> ERR_ACCESS_DENIED  equ  5 ; 'access denied !' ; TRDOS 8086
                        <1> ; 28/02/2017
                        <1> ERR_PERM_DENIED        equ 11 ; 'permission denied !' error
                        <1> ; 18/05/2016
                        <1> ERR_MISC        equ 27 ; miscellaneous/other errors
                        <1> ; 15/10/2016
                        <1> ; TRDOS 8086 -> TRDOS 386 (0Bh -> 28)
                        <1> ERR_INV_FORMAT         equ 28 ; 'invalid format !' error
                        <1> ; TRDOS 8086 -> TRDOS 386 (0Dh -> 29)
                        <1> ERR_INV_DATA    equ 29 ; 'invalid data !' error
                        <1> ; TRDOS 8086 -> TRDOS 386 (0Eh -> 20)
                        <1> ERR_ZERO_LENGTH        equ 20  ; 'zero length !' error
                        <1> ; TRDOS 8086 -> TRDOS 386 (15h -> 17, 1Dh -> 18, 1Eh -> 17)
                        <1> ERR_DRV_NR_READ        equ 17 ; 'drive not ready or read error !'
                        <1> ERR_DRV_NR_WRITE   equ 18 ; 'drive not ready or write error !'
                        <1> ; 15/10/2016
                        <1> ERR_INV_PATH_NAME  equ 19 ; 'bad path name !' error
                        <1> ERR_BAD_CMD_ARG        equ  1 ; 'bad command argument !' ; TRDOS 8086
                        <1> ERR_INV_FNUMBER        equ  1 ; 'invalid function number !' ; TRDOS 8086
                        <1> ERR_BIG_FILE    equ  8 ; 'big file & out of memory ! ; TRDOS 8086
                        <1> ERR_BIG_DATA    equ  8 ; 'big data & out of memory ! ; TRDOS 8086
                        <1> ERR_CLUSTER     equ 35 ; 'cluster not available !' ; TRDOS 8086
                        <1> ERR_OUT_OF_MEMORY  equ  4 ; 'out of memory !'
                        <1>                         ; 'insufficient memory !'
                        <1> ERR_P_VIOLATION        equ 6 ; 'protection violation !'
                        <1> ERR_PAGE_FAULT         equ 224 ;'page fault !' ;0E0h
                        <1> ERR_SWP_DISK_READ            equ 40
                        <1> ERR_SWP_DISK_NOT_PRESENT   equ 41
                        <1> ERR_SWP_SECTOR_NOT_PRESENT equ 42
                        <1> ERR_SWP_NO_FREE_SPACE      equ 43
                        <1> ERR_SWP_DISK_WRITE         equ 44
                        <1> ERR_SWP_NO_PAGE_TO_SWAP    equ 45
                        <1> ; 10/04/2017
                        <1> ERR_BUFFER      equ 46  ; 'buffer error !'
                        <1> ; 28/08/2017 (20/08/2017)
                        <1> ERR_DMA                equ -1  ; DMA buffer (allocation/misc.) error!
                        <1>
                        <1> ; 26/08/2015
                        <1> ; 24/07/2015
                        <1> ; 24/06/2015
                        <1> MAX_ARG_LEN     equ 256 ; max. length of sys exec arguments
                        <1> ; 01/07/2015
                        <1> MAX_MSG_LEN     equ 255 ; max. msg length for 'sysmsg'
                        <1> ;
                        <1> ; 06/10/2016
                        <1> OPENFILES       equ 10  ; max. number of open files (system)
                        <1> ; 07/10/2016
                        <1> ;NUMOFDEVICES          equ 20  ; max. num of available devices (sys)
                        <1>
                            %include 'trdosk0.s' ; 04/01/2016
                        <1> ; ****************************************************************************
                        <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DEFINITIONS : trdosk0.s
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Last Update: 29/02/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Beginning: 04/01/2016
                        <1> ; ----------------------------------------------------------------------------
                        <1> ; Assembler: NASM version 2.11 (trdos386.s)
```

```
<1> ; ----------------------------------------------------------------------------
<1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
<1> ; TRDOS2.ASM (09/11/2011)
<1> ; ****************************************************************************
<1> ; TRDOS2.ASM (c) 2004-2011 Erdogan TAN [ 17/01/2004 ] Last Update: 09/11/2011
<1> ;
<1> ; Masterboot / Partition Table at Beginning+1BEh
<1> ptBootable       equ 0
<1> ptBeginHead      equ 1
<1> ptBeginSector    equ 2
<1> ptBeginCylinder  equ 3
<1> ptFileSystemID   equ 4
<1> ptEndHead        equ 5
<1> ptEndSector      equ 6
<1> ptEndCylinder    equ 7
<1> ptStartSector    equ 8
<1> ptSectors        equ 12
<1>
<1> ; Boot Sector Parameters at 7C00h
<1> DataArea1     equ -4
<1> DataArea2     equ -2
<1> BootStart     equ 0h
<1> OemName       equ 03h
<1> BytesPerSec   equ 0Bh
<1> SecPerClust   equ 0Dh
<1> ResSectors    equ 0Eh
<1> FATs          equ 10h
<1> RootDirEnts   equ 11h
<1> Sectors       equ 13h
<1> Media         equ 15h
<1> FATSecs       equ 16h
<1> SecPerTrack   equ 18h
<1> Heads         equ 1Ah
<1> Hidden1       equ 1Ch
<1> Hidden2       equ 1Eh
<1> HugeSec1      equ 20h
<1> HugeSec2      equ 22h
<1> DriveNumber   equ 24h
<1> Reserved1     equ 25h
<1> bootsignature equ 26h
<1> VolumeID      equ 27h
<1> VolumeLabel   equ 2Bh
<1> FileSysType   equ 36h
<1> Reserved2     equ 3Eh                           ; Starting cluster of P2000
<1>
<1> ; FAT32 BPB Structure
<1> FAT32_FAT_Size equ 36
<1> FAT32_RootFClust equ 44
<1> FAT32_FSInfoSec equ 48
<1> FAT32_DrvNum equ 64
<1> FAT32_BootSig equ 66
<1> FAT32_VolID equ 67
<1> FAT32_VolLab equ 71
<1> FAT32_FilSysType equ 82
<1>
<1> ; BIOS Disk Parameters
<1> DPDiskNumber  equ 0h
<1> DPDType       equ 1h
<1> DPReturn      equ 2h
<1> DPHeads       equ 3h
<1> DPCylinders   equ 4h
<1> DPSecPerTrack equ 6h
<1> DPDisks       equ 7h
<1> DPTableOff    equ 8h
<1> DPTableSeg    equ 0Ah
<1> DPNumOfSecs   equ 0Ch
<1>
<1> ; BIOS INT 13h Extensions (LBA extensions)
<1> ; Just After DP Data (DPDiskNumber+)
<1> DAP_PacketSize equ 10h  ; If extensions present, this byte will be >=10h
<1> DAP_Reserved1 equ 11h   ; Reserved Byte
<1> DAP_NumOfBlocks equ 12h ; Value of this byte must be 0 to 127
<1> DAP_Reserved2 equ 13h   ; Reserved Byte
<1> DAP_Destination equ 14h ; Address of Transfer Buffer as SEGMENT:OFFSET
<1> DAP_LBA_Address equ 18h ; LBA=(C1*H0+H1)*S0+S1-1
<1>                         ; C1= Selected Cylinder Number
<1>                         ; H0= Number Of Heads (Maximum Head Number + 1)
<1>                         ; H1= Selected Head Number
<1>                         ; S0= Maximum Sector Number
<1>                         ; S1= Selected Sector Number
<1>                         ; QUAD WORD
<1> ; DAP_Flat_Destination equ 20h ; 64 bit address, if value in 4h is FFFF:FFFFh
<1>                               ; QUAD WORD (Also, value in 0h must be 18h)
<1>                               ; TR-DOS will not use 64 bit Flat Address
<1>
<1> ; INT 13h Function 48h "Get Enhanced Disk Drive Parameters"
<1> ; Just After DP Data (DPDiskNumber+)
<1> GetDParams_48h equ 20h ; Word. Data Length, must be 26 (1Ah) for short data.
<1> GDP_48h_InfoFlag equ 22h ; Word
<1> ; Bit 1 = 1 -> The geometry returned in bytes 4-15 is valid.
<1> GDP_48h_NumOfPCyls equ 24h ; Double Word. Number physical cylinders.
<1> GDP_48h_NumOfPHeads equ 28h ; Double Word. Number of physical heads.
<1> GDP_48h_NumOfPSpT equ 2Ch ; Double word. Num of physical sectors per track.
<1> GDP_48h_LBA_Sectors equ 30h ; 8 bytes. Number of physical/LBA sectors.
<1> GDP_48h_BytesPerSec equ 38h ; Word. Number of bytes in a sector.
<1>
<1> ; TR-DOS Standalone Program Extensions to the DiskParams Block
<1> ; Just After DP Data (DPDiskNumber+)
<1> TRDP_CurrentSector equ 3Ah  ; DX:AX (LBA)
<1> TRDP_SectorCount equ 3Eh    ; CX (or Counter)
<1>
<1>
<1> ; DOS Logical Disks
<1> LD_Name equ 0
<1> LD_DiskType equ 1
```

```
                        <1> LD_PhyDrvNo equ 2
                        <1> LD_FATType equ 3
                        <1> LD_FSType equ 4
                        <1> LD_LBAYes equ 5
                        <1> LD_BPB equ 6
                        <1> LD_FATBegin equ 96
                        <1> LD_ROOTBegin equ 100
                        <1> LD_DATABegin equ 104
                        <1> LD_StartSector equ 108
                        <1> LD_TotalSectors equ 112
                        <1> LD_FreeSectors equ 116
                        <1> LD_Clusters equ 120
                        <1> LD_PartitionEntry equ 124
                        <1> LD_DParamEntry equ 125
                        <1> LD_MediaChanged equ 126
                        <1> LD_CDirLevel equ 127
                        <1> LD_CurrentDirectory equ 128
                        <1>
                        <1> ; Singlix FS Extensions to DOS Logical Disks
                        <1> ; 03/01/2010 (LD_BPB compatibility for CHS r/w)
                        <1>
                        <1> LD_FS_Name equ 0
                        <1> LD_FS_DiskType equ 1
                        <1> LD_FS_PhyDrvNo equ 2
                        <1> LD_FS_FATType equ 3
                        <1> LD_FS_FSType equ 4
                        <1> LD_FS_LBAYes equ 5
                        <1> LD_FS_BPB equ 6
                        <1> LD_FS_MediaAttrib equ 6
                        <1> LD_FS_VersionMajor equ 7
                        <1> LD_FS_RootDirD equ 8
                        <1> LD_FS_MATLocation equ 12
                        <1> LD_FS_Reserved1 equ 16 ;1 reserved byte
                        <1> LD_FS_BytesPerSec equ 17 ; LD_BPB + 0Bh
                        <1> LD_FS_Reserved2 equ 19 ;2 reserved byte
                        <1> LD_FS_DATLocation equ 20
                        <1> LD_FS_DATSectors equ 24
                        <1> LD_FS_Reserved3 equ 28 ;3 reserved word
                        <1> LD_FS_SecPerTrack equ 30 ; LD_BPB + 18h
                        <1> LD_FS_NumHeads equ 32    ; LD_BPB + 1Ah
                        <1> LD_FS_UnDelDirD equ 34
                        <1> LD_FS_Reserved4 equ 38 ;4 reserved word
                        <1> LD_FS_VolumeSerial equ 40
                        <1> LD_FS_VolumeName equ 44
                        <1> LD_FS_BeginSector equ 108
                        <1> LD_FS_VolumeSize equ 112
                        <1> LD_FS_FreeSectors equ 116
                        <1> LD_FS_FirstFreeSector equ 120
                        <1> LD_FS_PartitionEntry equ 124
                        <1> LD_FS_DParamEntry equ 125
                        <1> LD_FS_MediaChanged equ 126
                        <1> LD_FS_CDirLevel equ 127
                        <1> LD_FS_CDIR_Converted equ 128
                        <1>
                        <1> ; Valid FAT Types
                        <1> FS_FAT12 equ 1
                        <1> FS_FAT16_CHS equ 2
                        <1> FS_FAT32_CHS equ 3
                        <1> FS_FAT16_LBA equ 4
                        <1> FS_FAT32_LBA equ 5
                        <1>
                        <1> ; Cursor Location
                        <1> CCCpointer equ  0450h   ; BIOS data, current cursor column
                        <1> ; FAT Clusters EOC sign
                        <1> FAT12EOC equ 0FFFh
                        <1> FAT16EOC equ 0FFFFh
                        <1> ;FAT32EOC equ 0FFFFFFFh ; It is not direct usable for 8086 code
                        <1> ; BAD Cluster
                        <1> FAT12BADC equ 0FF7h
                        <1> FAT16BADC equ 0FFF7h
                        <1> ;FAT32BADC equ 0FFFFFF7h ; It is not direct usable for 8086 code
                        <1> ; MS-DOS FAT16 FS (Maximum Possible) Last Cluster Number= 0FFF6h
                        <1>
                        <1> ; TRFS
                        <1>
                        <1> bs_FS_JmpBoot equ 0 ; jmp short bsBootCode
                        <1>                     ; db 0EBh, db 3Fh, db 90h
                        <1> bs_FS_Identifier equ 3  ; db 'FS', db 0
                        <1> bs_FS_BytesPerSec equ 6 ; dw 512
                        <1> bs_FS_MediaAttrib equ 8 ; db 3
                        <1> bs_FS_PartitionID equ 9 ; db 0A1h
                        <1> bs_FS_VersionMaj equ 10 ; db 01h
                        <1> bs_FS_VersionMin equ 11 ; db 0
                        <1> bs_FS_BeginSector equ 12   ; dd 0
                        <1> bs_FS_VolumeSize equ 16 ; dd 2880
                        <1> bs_FS_StartupFD equ 20 ; dd 0
                        <1> bs_FS_MATLocation equ 24 ; dd 1
                        <1> bs_FS_RootDirD equ 28 ; dd 8
                        <1> bs_FS_SystemConfFD equ 32 ; dd 0
                        <1> bs_FS_SwapFD equ 36 ; dd 0
                        <1> bs_FS_UnDelDirD equ 40 ; dd 0
                        <1> bs_FS_DriveNumber equ 44 ; db 0
                        <1> bs_FS_LBA_Ready equ 45 ; db 0
                        <1> bs_FS_MagicWord equ 46
                        <1> bs_FS_SecPerTrack equ 46 ; db 0A1h
                        <1> bs_FS_Heads equ 47 ; db 01h
                        <1> bs_FS_OperationSys equ 48 ; db "TR-SINGLIX v1.0b"
                        <1> bs_FS_Terminator equ 64 ; db 0
                        <1> bs_FS_BootCode equ 65
                        <1>
                        <1> FS_MAT_DATLocation equ 12
                        <1> FS_MAT_DATScount equ 16
                        <1> FS_MAT_FreeSectors equ 20
                        <1> FS_MAT_FirstFreeSector equ 24
                        <1> FS_RDT_VolumeSerialNo equ 28
```

```
                                <1> FS_RDT_VolumeName equ 64
                                <1>
                                <1> ; FAT12 + FAT16 + FAT32
                                <1> BS_JmpBoot equ 0
                                <1> BS_OEMName equ 3
                                <1> BPB_BytsPerSec equ 11
                                <1> BPB_SecPerClust equ 13
                                <1> BPB_RsvdSecCnt equ 14
                                <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
                                    %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)
00007446 B036                   <1>       mov   al, 00110110b ; 36h
00007448 E643                   <1>       out   43h, al
0000744A 31C0                   <1>       xor   eax, eax  ; sub     al, al ; 0
0000744C E640                   <1>       out   40h, al ; LB
0000744E E640                   <1>       out   40h, al ; HB
                                <1>       ;
                                <1>       ; 30/03/2016
                                <1>       ; Clear Logical DOS Disk Description Tables Area
                                <1>       ;xor   eax, eax
```

```
00007450 BF00010900          <1>       mov    edi, Logical_DOSDisks
00007455 B980060000          <1>       mov    ecx, 6656/4 ; 26*256 = 6656 bytes
0000745A F3AB                <1>       rep    stosd ; 1664 times 4 bytes
                             <1>
0000745C B83F3A2F00          <1>       mov    eax, '?:/'
00007461 A3[DF8A0100]        <1>       mov    [Current_Dir_Drv], eax
                             <1>
                             <1>       ; Logical DRV INIT (only for hard disks)
00007466 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>
0000746B BE00010900          <1>       mov    esi, Logical_DOSDisks
00007470 B001                <1>       mov    al, 1 ; Initialization sign (invalid_fd_parameter)
00007472 83C67E              <1>       add    esi, LD_MediaChanged ; Media Change Status = 1 (init needed)
00007475 8806                <1>       mov    [esi], al ; A:
00007477 81C600010000        <1>       add    esi, 100h
0000747D 8806                <1>       mov    [esi], al ; B:
                             <1>
                             <1> _current_drive_bootdisk:
0000747F 8A15[0A6E0000]      <1>       mov    dl, [boot_drv] ; physical drive number
00007485 80FAFF              <1>       cmp    dl, 0FFh
00007488 740A                <1>       je     short _last_dos_diskno_check
                             <1> _boot_drive_check:
0000748A 80FA80              <1>       cmp    dl, 80h
0000748D 7218                <1>       jb     short _current_drive_a
0000748F 80EA7E              <1>       sub    dl, 7Eh ; C = 2 , D = 3
00007492 EB13                <1>       jmp    short _current_drive_a
                             <1>
                             <1> _last_dos_diskno_check:
00007494 8A15[A5400100]      <1>       mov    dl, [Last_DOS_DiskNo]
0000749A 80FA02              <1>       cmp    dl, 2
0000749D 7706                <1>       ja     short _current_drive_c
0000749F 7406                <1>       je     short _current_drive_a
000074A1 30D2                <1>       xor    dl, dl ; A:
000074A3 EB02                <1>       jmp    short _current_drive_a
                             <1>
                             <1> _current_drive_c:
000074A5 B202                <1>       mov    dl, 2 ; C:
                             <1>
                             <1> _current_drive_a:
000074A7 8815[0B6E0000]      <1>       mov    [drv], dl
000074AD BE[A7400100]        <1>         mov     esi, msg_CRLF_temp
000074B2 E8AE000000          <1>       call   print_msg
                             <1>
000074B7 8A15[0B6E0000]      <1>       mov    dl, [drv]
                             <1> _default_drive_c:
000074BD E82C0C0000          <1>       call   change_current_drive
000074C2 731C                <1>       jnc    short _start_mainprog
                             <1>
                             <1> _drv_not_ready_error:
000074C4 BE[62430100]        <1>       mov    esi, msgl_drv_not_ready
000074C9 E897000000          <1>       call   print_msg
                             <1>         ;jmp _end_of_mainprog
                             <1>
                             <1>       ; 20/01/2018
000074CE B202                <1>       mov    dl, 2
000074D0 3815[0B6E0000]      <1>       cmp    [drv], dl
000074D6 736B                <1>       jnb    short _end_of_mainprog
000074D8 8815[0B6E0000]      <1>       mov    [drv], dl
000074DE 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
000074E0 66B80100            <1>       mov    ax, 1
000074E4 A2[B3030300]        <1>       mov    [u.uno], al
000074E9 66A3[4E030300]      <1>       mov    [mpid], ax
000074EF 66A3[20000300]      <1>       mov    [p.pid], ax
000074F5 A2[B0000300]        <1>       mov    [p.stat], al
000074FA C605[A8030300]04    <1>       mov    byte [u.quant], time_count  ; 07/01/2017
                             <1>       ;
00007501 A1[188A0100]        <1>       mov    eax, [k_page_dir]
00007506 A3[B8030300]        <1>       mov    [u.pgdir], eax ; reset
                             <1>       ;
0000750B E8CBE6FFFF          <1>       call   allocate_page
00007510 0F82B5000000        <1>       jc     panic
00007516 A3[B4030300]        <1>       mov    [u.upage], eax ; user structure page
0000751B A3[C0000300]        <1>       mov    [p.upage], eax
00007520 E830E7FFFF          <1>       call   clear_page
                             <1>       ;
                             <1>       ; 24/08/2015
00007525 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
0000752B BF00300900          <1>       mov    edi, Env_Page ; 93000h
00007530 B980000000          <1>       mov    ecx, Env_Page_Size / 4    ; 512/4  (4096/4)

00007535 31C0                <1>       xor    eax, eax
00007537 F3AB                <1>       rep    stosd
                             <1>
                             <1>       ; 14/04/2016
00007539 E807350000          <1>       call   mainprog_startup_configuration
                             <1>
0000753E E8EC0C0000          <1>         call    dos_prompt
```

```
                            <1>
                            <1> _end_of_mainprog:
00007543 BE[A7400100]       <1>         mov     esi, msg_CRLF_temp
00007548 E818000000         <1>       call   print_msg
0000754D BE[AD400100]       <1>       mov    esi, mainprog_Version
00007552 E80E000000         <1>       call   print_msg
                            <1>       ; 24/01/2016
00007557 28E4               <1>       sub    ah, ah
00007559 E88F99FFFF         <1>       call   int16h ; call getch
0000755E E96F9EFFFF         <1>       jmp    cpu_reset
                            <1>
00007563 EBFE               <1> infinitiveloop: jmp short infinitiveloop
                            <1>
                            <1> print_msg:
                            <1>       ; 13/05/2016
                            <1>       ; 04/01/2016
                            <1>       ; 01/07/2015
                            <1>       ; 13/03/2015 (Retro UNIX 386 v1)
                            <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
                            <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI, EDI)
                            <1>       ;
00007565 8A3D[468A0100]     <1>       mov    bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
                            <1>       ;mov   bl, 07h ; Black background, light gray forecolor
                            <1>
0000756B AC                 <1>       lodsb
                            <1> pmsg1:
0000756C 56                 <1>       push   esi
                            <1>       ;mov   bh, [ACTIVE_PAGE] ; 04/01/2016 (ptty)
0000756D B307               <1>       mov    bl, 07h ; Black background, light gray forecolor
0000756F E891ADFFFF         <1>       call   _write_tty
00007574 5E                 <1>       pop    esi
00007575 AC                 <1>       lodsb
00007576 20C0               <1>       and    al, al
00007578 75F2               <1>       jnz    short pmsg1
0000757A C3                 <1>       retn
                            <1>
                            <1> clear_screen:
                            <1>       ; 06/12/2020
                            <1>       ; 03/12/2020 (TRDOS 386 v2.0.3)
                            <1>       ; 13/05/2016
                            <1>       ; 30/01/2016
                            <1>       ; 24/01/2016
                            <1>       ; 04/01/2016
0000757B 0FB61D[468A0100]   <1>       movzx  ebx, byte [ACTIVE_PAGE] ; video page number (0 to 7)
00007582 8AA3[EB6F0000]     <1>       mov    ah, [ebx+vmode] ; default = 03h (80x25 text)
00007588 80FC04             <1>       cmp    ah, 4
0000758B 7205               <1>       jb     short cls1
0000758D 80FC07             <1>       cmp    ah, 7
00007590 7530               <1>       jne    short vga_clear
                            <1> cls1:
                            <1>       ;mov   bh, bl
                            <1>       ;mov   bl, 7
00007592 3A25[DA6F0000]     <1>       cmp    ah, [CRT_MODE] ; current video mode ?
00007598 740E               <1>       je     short cls2 ; yes (current video mode = 3)
                            <1>       ;;call set_mode_3 ; set video mode to 3 (& clear screen)
                            <1>       ;;retn
                            <1>       ; 06/12/2020
0000759A 803D[1C120300]00   <1>       cmp    byte [pmi32], 0
000075A1 771F               <1>       ja     short vga_clear
000075A3 E967ADFFFF         <1>       jmp    set_mode_3
                            <1> cls2:
000075A8 88DF               <1>       mov    bh, bl ; video page (0 to 7)
000075AA B307               <1>       mov    bl, 07h ; attribute to be used on blanked line
000075AC 28C0               <1>       sub    al, al ; 0 =  entire window
000075AE 6631C9             <1>       xor    cx, cx
000075B1 66BA4F18           <1>       mov    dx, 184Fh
000075B5 E88BAAFFFF         <1>       call   _scroll_up ; 24/01/2016
                            <1>       ;
                            <1>       ;mov   bh, [ACTIVE_PAGE] ; video page number (0 to 7)
000075BA 6631D2             <1>       xor    dx, dx
                            <1>       ;call  _set_cpos ; 24/01/2016
                            <1>       ;;retn
                            <1>       ; 03/12/2020
000075BD E9DDADFFFF         <1>       jmp    _set_cpos ; returns to the caller of this proc
                            <1> ;cls3:
                            <1> ;     retn
                            <1>       ; 06/12/2020
                            <1> vga_clear:
                            <1>       ; 03/12/2020
                            <1>       ; set mode by using _int10h
                            <1>       ; (also clears screen)
000075C2 88E0               <1>       mov    al, ah
000075C4 28E4               <1>       sub    ah, ah  ; set current video mode
                            <1>       ;call  _int10h ; simulates int 10h in TRDOS 386 kernel
                            <1>       ;jmp   short cls3
000075C6 E9A6A1FFFF         <1>       jmp    _int10h ; returns to the caller of this proc
                            <1>
                            <1> panic:
                            <1>       ; 13/05/2016 (TRDOS 386 = TRDOS v2)
                            <1>       ; 13/03/2015 (Retro UNIX 386 v1)
                            <1>       ; 07/03/2014 (Retro UNIX 8086 v1)
000075CB BE[6B4D0100]       <1>       mov    esi, panic_msg
000075D0 E890FFFFFF         <1>       call   print_msg
                            <1> key_to_reboot:
                            <1>         ; 24/01/2016
000075D5 28E4               <1>         sub     ah, ah
000075D7 E81199FFFF         <1>         call    int16h ; call   getch
                            <1>         ; wait for a character from the current tty
                            <1>       ;
000075DC B00A               <1>       mov    al, 0Ah
000075DE 8A3D[468A0100]     <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
000075E4 B307               <1>       mov    bl, 07h ; Black background,
                            <1>                    ; light gray forecolor
000075E6 E81AADFFFF         <1>       call   _write_tty
000075EB E9E29DFFFF         <1>       jmp    cpu_reset
```

```
                            <1>
                            <1> ctrlbrk:
                            <1>       ; 12/11/2015
                            <1>       ; 13/03/2015 (Retro UNIX 386 v1)
                            <1>       ; 06/12/2013 (Retro UNIX 8086 v1)
                            <1>       ;
                            <1>       ; INT 1Bh (control+break) handler
                            <1>       ;
                            <1>             ; Retro Unix 8086 v1 feature only!
                            <1>             ;
000075F0 66833D[AA030300]00 <1>       cmp    word [u.intr], 0
000075F8 7645               <1>       jna    short cbrk4
                            <1> cbrk0:
                            <1>       ; 12/11/2015
                            <1>       ; 06/12/2013
000075FA 66833D[AC030300]00 <1>       cmp    word [u.quit], 0
00007602 743B               <1>       jz     short cbrk4
                            <1>       ;
                            <1>       ; 20/09/2013
00007604 6650               <1>       push   ax
00007606 A0[468A0100]       <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>       ;
0000760B 3C08               <1>       cmp    al, 8 ; serial port tty nums > 7
0000760D 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>       ;
0000760F 52                 <1>       push   edx
00007610 0FB615[B3030300]   <1>       movzx  edx, byte [u.uno]
00007617 3A82[7F000300]     <1>       cmp     al, [edx+p.ttyc-1] ; console tty (rw)
0000761D 5A                 <1>       pop    edx
0000761E 7412               <1>       je     short cbrk2
                            <1> cbrk1:
00007620 FEC0               <1>       inc    al  ; [u.ttyp] : 1 based tty number
                            <1>       ; 06/12/2013
00007622 3A05[94030300]     <1>       cmp    al, [u.ttyp] ; recent open tty (r)
00007628 7408               <1>       je     short cbrk2
0000762A 3A05[95030300]     <1>         cmp     al, [u.ttyp+1] ; recent open tty (w)
00007630 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>       ;
00007632 6631C0             <1>       xor    ax, ax ; 0
00007635 6648               <1>       dec    ax
                            <1>       ; 0FFFFh = 'ctrl+brk' keystroke
00007637 66A3[AC030300]     <1>       mov    [u.quit], ax
                            <1> cbrk3:
0000763D 6658               <1>       pop    ax
                            <1> cbrk4:
0000763F 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>
00007640 E86DF4FFFF         <1>       call   get_rtc_time       ; Return Current Time
00007645 86E9               <1>         xchg       ch,cl
00007647 66890D[E2860100]   <1>         mov  [hour], cx
0000764E 86F2               <1>         xchg       dh,dl
00007650 668915[E6860100]   <1>         mov  [second], dx
                            <1>       ;
00007657 E8C7F4FFFF         <1>         call       get_rtc_date       ; Return Current Date
0000765C 86E9               <1>         xchg       ch,cl
0000765E 66890D[DC860100]   <1>         mov  [year], cx
00007665 86F2               <1>         xchg       dh,dl
```

```
00007667 668915[DE860100]    <1>         mov  [month], dx
                             <1>       ;
0000766E 66B93030            <1>       mov    cx, 3030h
                             <1>       ;
00007672 A0[E2860100]        <1>       mov    al, [hour] ; Hour
                             <1>              ; AL <= BCD number)
00007677 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
00007679 D50A                <1>         aad ; AX= AH*10+AL
0000767B A2[E2860100]        <1>       mov    [hour], al
00007680 A0[E3860100]        <1>       mov    al, [hour+1] ; Minute
                             <1>              ; AL <= BCD number)
00007685 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
00007687 D50A                <1>         aad ; AX= AH*10+AL
00007689 A2[E4860100]        <1>       mov    [minute], al
0000768E A0[E6860100]        <1>       mov    al, [second] ; Second
                             <1>              ; AL <= BCD number)
00007693 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
00007695 D50A                <1>         aad ; AX= AH*10+AL
00007697 A2[E6860100]        <1>       mov    [second], al
0000769C 66A1[DC860100]      <1>       mov    ax, [year] ; Year (century)
000076A2 6650                <1>         push       ax
                             <1>              ; AL <= BCD number)
000076A4 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
000076A6 D50A                <1>         aad ; AX= AH*10+AL
000076A8 B464                <1>       mov    ah, 100
000076AA F6E4                <1>       mul    ah
000076AC 66A3[DC860100]      <1>       mov    [year], ax
000076B2 6658                <1>       pop    ax
000076B4 88E0                <1>       mov    al, ah
                             <1>              ; AL <= BCD number)
000076B6 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
000076B8 D50A                <1>         aad ; AX= AH*10+AL
000076BA 660105[DC860100]    <1>       add    [year], ax
000076C1 A0[DE860100]        <1>       mov    al, [month] ; Month
                             <1>              ; AL <= BCD number)
000076C6 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
000076C8 D50A                <1>         aad ; AX= AH*10+AL
000076CA A2[DE860100]        <1>       mov    [month], al
000076CF A0[DF860100]        <1>         mov     al, [month+1]          ; Day
                             <1>              ; AL <= BCD number)
000076D4 D410                <1>         db   0D4h,10h            ; Undocumented inst. AAM
                             <1>                                  ; AH = AL / 10h
                             <1>                                  ; AL = AL MOD 10h
000076D6 D50A                <1>         aad ; AX= AH*10+AL
000076D8 A2[E0860100]        <1>         mov     [day], al
                             <1>
000076DD C3                  <1>       retn   ; 30/12/2017
                             <1>
                             <1> epoch:
000076DE E85DFFFFFF          <1>       call   get_rtc_date_time ; TRDOS 386 - 30/12/2017
                             <1>
                             <1> convert_to_epoch:
                             <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
                             <1>       ; 15/03/2015 (Retro UNIX 386 v1 - 32 bit modification)
                             <1>       ; 09/04/2013 (Retro UNIX 8086 v1)
                             <1>       ;
                             <1>       ; ((Modified registers: EAX, EDX, EBX))
                             <1>       ;
                             <1>       ; Derived from DALLAS Semiconductor
                             <1>       ; Application Note 31 (DS1602/DS1603)
                             <1>       ; 6 May 1998
000076E3 29C0                <1>       sub    eax, eax
000076E5 66A1[DC860100]      <1>       mov    ax, [year]
000076EB 662DB207            <1>       sub    ax, 1970
000076EF BA6D010000          <1>       mov    edx, 365
000076F4 F7E2                <1>       mul    edx
000076F6 31DB                <1>       xor    ebx, ebx
000076F8 8A1D[DE860100]      <1>       mov    bl, [month]
000076FE FECB                <1>       dec    bl
00007700 D0E3                <1>       shl    bl, 1
                             <1>       ;sub   edx, edx
00007702 668B93[E8860100]    <1>       mov    dx, [EBX+DMonth]
00007709 8A1D[E0860100]      <1>         mov     bl, [day]
0000770F FECB                <1>       dec    bl
00007711 01D0                <1>       add    eax, edx
00007713 01D8                <1>       add    eax, ebx
                             <1>                     ; EAX = days since 1/1/1970
00007715 668B15[DC860100]    <1>       mov    dx, [year]
0000771C 6681EAB107          <1>       sub    dx, 1969
00007721 66D1EA              <1>       shr    dx, 1
00007724 66D1EA              <1>       shr    dx, 1
                             <1>              ; (year-1969)/4
00007727 01D0                <1>       add    eax, edx
                             <1>                     ; + leap days since 1/1/1970
00007729 803D[DE860100]02    <1>       cmp    byte [month], 2     ; if past february
00007730 7610                <1>       jna    short cte1
00007732 668B15[DC860100]    <1>       mov    dx, [year]
00007739 6683E203            <1>       and    dx, 3 ; year mod 4
0000773D 7503                <1>       jnz    short cte1
                             <1>                     ; and if leap year
0000773F 83C001              <1>       add    eax, 1       ; add this year's leap day (february 29)
                             <1> cte1:                     ; compute seconds since 1/1/1970
00007742 BA18000000          <1>       mov    edx, 24
```

```
00007747 F7E2                <1>       mul    edx
00007749 8A15[E2860100]      <1>       mov    dl, [hour]
0000774F 01D0                <1>       add    eax, edx
                             <1>              ; EAX = hours since 1/1/1970 00:00:00
                             <1>       ;mov   ebx, 60
00007751 B33C                <1>       mov    bl, 60
00007753 F7E3                <1>       mul    ebx
00007755 8A15[E4860100]      <1>       mov    dl, [minute]
0000775B 01D0                <1>       add    eax, edx
                             <1>              ; EAX = minutes since 1/1/1970 00:00:00
                             <1>       ;mov   ebx, 60
0000775D F7E3                <1>       mul    ebx
0000775F 8A15[E6860100]      <1>       mov    dl, [second]
00007765 01D0                <1>       add    eax, edx
                             <1>              ; EAX -> seconds since 1/1/1970 00:00:00
00007767 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>       ;
00007768 31D2                <1>       xor    edx, edx
0000776A B93C000000          <1>       mov    ecx, 60
0000776F F7F1                <1>       div    ecx
                             <1>       ;mov   [imin], eax   ; whole minutes
                             <1>                            ; since 1/1/1970
00007771 668915[E6860100]    <1>       mov    [second], dx  ; leftover seconds
00007778 29D2                <1>       sub    edx, edx
0000777A F7F1                <1>       div    ecx
                             <1>       ;mov   [ihrs], eax   ; whole hours
                             <1>       ;                   ; since 1/1/1970
0000777C 668915[E4860100]    <1>       mov    [minute], dx  ; leftover minutes
00007783 31D2                <1>       xor    edx, edx
                             <1>       ;mov   cx, 24
00007785 B118                <1>       mov    cl, 24
00007787 F7F1                <1>       div    ecx
                             <1>       ;mov   [iday], ax   ; whole days
                             <1>                           ; since 1/1/1970
00007789 668915[E2860100]    <1>       mov    [hour], dx   ; leftover hours
00007790 05DB020000          <1>       add    eax, 365+366 ; whole day since
                             <1>                           ; 1/1/1968
                             <1>       ;mov   [iday], ax
00007795 50                  <1>       push   eax
00007796 29D2                <1>       sub    edx, edx
00007798 B9B5050000          <1>       mov    ecx, (4*365)+1 ; 4 years = 1461 days
0000779D F7F1                <1>       div    ecx
0000779F 59                  <1>       pop    ecx
                             <1>       ;mov   [lday], ax   ; count of quadyrs (4 years)
000077A0 6652                <1>       push   dx
                             <1>       ;mov   [qday], dx   ; days since quadyr began
000077A2 6683FA3C            <1>       cmp    dx, 31 + 29  ; if past feb 29 then
000077A6 F5                  <1>       cmc                 ; add this quadyr's leap day
000077A7 83D000              <1>       adc    eax, 0       ; to # of qadyrs (leap days)
                             <1>       ;mov   [lday], ax   ; since 1968
                             <1>       ;mov   cx, [iday]
000077AA 91                  <1>       xchg   ecx, eax     ; ECX = lday, EAX = iday
000077AB 29C8                <1>       sub    eax, ecx     ; iday - lday
000077AD B96D010000          <1>       mov    ecx, 365
000077B2 31D2                <1>       xor    edx, edx
                             <1>       ; EAX = iday-lday, EDX = 0
000077B4 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
000077B6 6605B007            <1>       add    ax, 1968     ; compute year
000077BA 66A3[DC860100]      <1>       mov    [year], ax
000077C0 6689D1              <1>       mov    cx, dx
                             <1>       ;mov   dx, [qday]
000077C3 665A                <1>       pop    dx
000077C5 6681FA6D01          <1>       cmp    dx, 365           ; if qday <= 365 and qday >= 60
000077CA 7709                <1>       ja     short cfe1   ; jday = jday +1
000077CC 6683FA3C            <1>       cmp    dx, 60       ; if past 2/29 and leap year then
000077D0 F5                  <1>         cmc              ; add a leap day to the # of whole
000077D1 6683D100            <1>       adc    cx, 0        ; days since 1/1 of current year
                             <1> cfe1:
                             <1>       ;mov   [jday], cx
000077D5 66BB0C00            <1>       mov    bx, 12       ; estimate month
000077D9 66BA6E01            <1>       mov    dx, 366      ; mday, max. days since 1/1 is 365
000077DD 6683E003            <1>       and    ax, 11b      ; year mod 4 (and dx, 3)
                             <1> cfe2: ; Month calculation  ; 0 to 11  (11 to 0)
000077E1 6639D1              <1>       cmp    cx, dx       ; mday = # of days passed from 1/1
000077E4 731D                <1>       jnb    short cfe3
000077E6 664B                <1>       dec    bx           ; month = month - 1
000077E8 66D1E3              <1>       shl    bx, 1
000077EB 668B93[E8860100]    <1>       mov    dx, [EBX+DMonth] ; # elapsed days at 1st of month
000077F2 66D1EB              <1>       shr    bx, 1        ; bx = month - 1 (0 to 11)
000077F5 6683FB01            <1>       cmp    bx, 1        ; if month > 2 and year mod 4  = 0
000077F9 76E6                <1>       jna    short cfe2   ; then mday = mday + 1
000077FB 08C0                <1>       or     al, al       ; if past 2/29 and leap year then
000077FD 75E2                <1>       jnz    short cfe2   ; add leap day (to mday)
```

```
000077FF 6642                <1>       inc    dx           ; mday = mday + 1
00007801 EBDE                <1>       jmp    short cfe2
                             <1> cfe3:
00007803 6643                <1>       inc    bx          ; -> bx = month, 1 to 12
00007805 66891D[DE860100]    <1>       mov    [month], bx
0000780C 6629D1              <1>       sub    cx, dx      ; day = jday - mday + 1
0000780F 6641                <1>       inc    cx
00007811 66890D[E0860100]    <1>       mov    [day], cx
                             <1>
                             <1>       ; eax, ebx, ecx, edx is changed at return
                             <1>       ; output ->
                             <1>       ; [year], [month], [day], [hour], [minute], [second]
                             <1>
00007818 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)
00007819 E80F000000          <1>       call   set_date_bcd
                             <1>       ; Set real-time clock date
0000781E E82DF3FFFF          <1>       call   set_rtc_date ; RTC_50
                             <1>       ; Set real-time clock time
00007823 E832000000          <1>       call   set_time_bcd
00007828 E9B4F2FFFF          <1>       jmp    set_rtc_time ; RTC_30
                             <1>
                             <1> ; 31/12/2017
                             <1> set_date_bcd:
0000782D A0[DD860100]        <1>         mov     al, [year+1]
00007832 D40A                <1>       aam    ; ah = al / 10, al = al mod 10
00007834 D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
00007836 88C5                <1>       mov    ch, al ; century (BCD)
00007838 A0[DC860100]        <1>       mov    al, [year]
0000783D D40A                <1>       aam    ; ah = al / 10, al = al mod 10
0000783F D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
00007841 88C1                <1>       mov    cl, al ; year (BCD)
00007843 A0[DE860100]        <1>         mov  al, [month]
00007848 D40A                <1>       aam    ; ah = al / 10, al = al mod 10
0000784A D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
0000784C 88C6                <1>       mov    dh, al ; month (BCD)
0000784E A0[E0860100]        <1>       mov    al, [day]
00007853 D40A                <1>       aam    ; ah = al / 10, al = al mod 10
00007855 D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
00007857 88C6                <1>       mov    dh, al ; day (BCD)
00007859 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)
0000785A FA                  <1>       cli
0000785B E842F4FFFF          <1>       CALL   UPD_IPR             ; CHECK FOR UPDATE IN PROCESS
                             <1>       ; cf = 1 -> al = 0
00007860 7207                <1>         jc      short stime1
00007862 B00B                <1>       MOV    AL,CMOS_REG_B       ; ADDRESS ALARM REGISTER
00007864 E854F4FFFF          <1>       CALL   CMOS_READ           ; READ CURRENT VALUE OF DSE BIT
                             <1> stime1:
00007869 FB                  <1>       sti
0000786A 2401                <1>       AND    AL,00000001B        ; MASK FOR VALID DSE BIT
0000786C 88C2                <1>       MOV    DL,AL               ; SET [DL] TO ZERO FOR NO DSE BIT
                             <1>       ; DL = 1 or 0 (day light saving time)
                             <1>       ;
0000786E A0[E2860100]        <1>       mov    al, [hour]
00007873 D40A                <1>       aam    ; ah = al / 10, al = al mod 10
00007875 D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
00007877 88C5                <1>       mov    ch, al ; hour (BCD)
00007879 A0[E4860100]        <1>         mov     al, [minute]
0000787E D40A                <1>       aam    ; ah = al / 10, al = al mod 10
00007880 D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
00007882 88C1                <1>       mov    cl, al       ; minute (BCD)
00007884 A0[E6860100]        <1>         mov     al, [second]
00007889 D40A                <1>       aam    ; ah = al / 10, al = al mod 10
0000788B D510                <1>       db     0D5h,10h     ; Undocumented inst. AAD
                             <1>                           ; AL = AH * 10h + AL
0000788D 88C6                <1>       mov    dh, al      ; second (BCD)
0000788F 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
                                <1>
                                <1>       ; 15/07/2020
                                <1>       ;movzx ecx, byte [HF_NUM] ; number of fixed disks
                                <1>       ;cmp   cl, 1
                                <1>       ;jnb   short load_hd_partition_tables
                                <1>
00007890 A0[B48A0100]           <1>       mov    al, [HF_NUM] ; number of fixed disks
00007895 20C0                   <1>       and    al, al
00007897 7501                   <1>       jnz    short load_hd_partition_tables
                                <1>
                                <1>       ; no any hard disks
00007899 C3                     <1>       retn
                                <1>
                                <1> load_hd_partition_tables:
                                <1>       ;mov   esi, [HDPM_TBL_VEC] ; primary master disk FDPT
                                <1>       ; 15/07/2020
0000789A BE[B88A0100]           <1>       mov    esi, HDPM_TBL_VEC
0000789F BF[DE8E0100]           <1>       mov    edi, PTable_hd0
000078A4 B280                   <1>       mov    dl, 80h
                                <1>       ; 15/07/2020
000078A6 A2[0D6E0000]           <1>       mov    [hdc], al
                                <1>       ;xor   ecx, ecx ; 0
                                <1> load_next_hd_partition_table:
                                <1>       ; 20/07/2020
000078AB 31C9                   <1>       xor    ecx, ecx ; 0
                                <1>       ;push  ecx
000078AD 57                     <1>       push   edi ; *
                                <1>       ;push  esi ; FDPT (+ DPTE) address
                                <1>       ; 15/07/2020
000078AE AD                     <1>       lodsd
000078AF 56                     <1>       push   esi ; ** ; next FDPT (+ DPTE) address ptr
                                <1>
                                <1>       ;mov   al, [esi+20] ; DPTE offset 4
                                <1>       ;and   al, 40h ;  LBA bit (bit 6)
                                <1>       ;;shr  al, 6
                                <1>       ;mov   [HD_LBA_yes], al
                                <1>
                                <1>       ; 15/07/2020
000078B0 8A4814                 <1>       mov    cl, [eax+20]
000078B3 80E140                 <1>       and    cl, 40h
000078B6 880D[E28F0100]         <1>       mov    [HD_LBA_yes], cl
                                <1>
000078BC E844040000             <1>       call   load_masterboot
                                <1>       ;jc    short pass_pt_this_hard_disk
                                <1>       ; 13/08/2020
000078C1 0F828A000000           <1>       jc     pass_pt_this_hard_disk
                                <1>
000078C7 BB[9C8E0100]           <1>       mov    ebx, PartitionTable
000078CC 89DE                   <1>       mov    esi, ebx
                                <1>       ;mov   ecx, 16
000078CE B110                   <1>       mov    cl, 16
000078D0 F3A5                   <1>       rep    movsd
000078D2 89DE                   <1>       mov    esi, ebx
                                <1>       ;mov   byte [hdc], 4 ; 4 - partition index
                                <1>       ; 15/07/2020
000078D4 C605[E38F0100]04       <1>       mov    byte [PP_Counter], 4
                                <1> loc_validate_hdp_partition:
                                <1>       ;cmp   byte [esi+ptFileSystemID], 0
                                <1>       ;jna   short loc_validate_next_hdp_partition2
                                <1>       ; 13/08/2020
000078DB 8A4604                 <1>       mov    al, [esi+ptFileSystemID]
000078DE 20C0                   <1>       and    al, al
000078E0 7457                   <1>       jz     short loc_validate_next_hdp_partition2
                                <1>
000078E2 56                     <1>       push   esi ; *** ; Masterboot partition table offset
000078E3 52                     <1>       push   edx ; **** ; dl = Physical drive number
                                <1>
                                <1>       ; 13/08/2020
000078E4 3C05                   <1>       cmp    al, 05h  ; Extended partition CHS
000078E6 7404                   <1>       je     short loc_set_ep_counter
000078E8 3C0F                   <1>       cmp    al, 0Fh  ; Extended partition LBA
000078EA 7511                   <1>       jne    short loc_validate_next_hdp_partition0
                                <1>
                                <1>       ;;inc  byte [PP_Counter]
                                <1>       ; 15/07/2020
                                <1>       ;inc   byte [EP_Counter] ; disk has valid partition(s)
                                <1>
                                <1> loc_set_ep_counter:
                                <1>       ; 13/08/2020
000078EC 803D[E48F0100]80       <1>       cmp    byte [EP_Counter], 80h
000078F3 7342                   <1>       jnb    short loc_validate_next_hdp_partition1
                                <1>
000078F5 8815[E48F0100]         <1>       mov    byte [EP_Counter], dl ; disk drv has extd. part.
                                <1>
000078FB EB3A                   <1>       jmp    short loc_validate_next_hdp_partition1
                                <1>
                                <1> loc_validate_next_hdp_partition0:
000078FD 31FF                   <1>       xor    edi, edi ; 0
                                <1>       ; Input -> ESI = PartitionTable offset
                                <1>       ; DL = Hard disk drive number
                                <1>       ; EDI = 0 -> Primary Partition
                                <1>       ; EDI > 0 -> Extended Partition's Start Sector
000078FF E885010000             <1>       call   validate_hd_fat_partition
00007904 730E                   <1>       jnc    short loc_set_valid_hdp_partition_entry
                                <1>
                                <1>       ;pop   edx
```

```
                                <1>       ;push  edx
00007906 8B1424                 <1>       mov    edx, [esp]  ; ****
00007909 8B742404               <1>       mov    esi, [esp+4] ; *** ; 30/01/2018
0000790D E8D1020000             <1>       call   validate_hd_fs_partition
00007912 7223                   <1>       jc     short loc_validate_next_hdp_partition1
                                <1> loc_set_valid_hdp_partition_entry:
00007914 8A0D[A5400100]         <1>       mov    cl, [Last_DOS_DiskNo]
0000791A 80C141                 <1>       add    cl, 'A'
                                <1>       ; ESI = Logical dos drive description table address
0000791D 880E                   <1>       mov    [esi+LD_Name], cl
                                <1>       ; 15/07/2020
0000791F 8A4602                 <1>       mov    al, [esi+LD_PhyDrvNo] ; Physical drive number
                                <1>       ;mov   al, [esp] ; ****
00007922 2C7F                   <1>       sub    al, 7Fh
                                <1>              ; AL = 1 to 4
00007924 C0E002                 <1>       shl    al, 2 ; AL = 4 to 16
                                <1>
00007927 8A15[E38F0100]         <1>       mov    dl, [PP_Counter]
                                <1>
                                <1>       ;sub   al, [PP_Counter]
0000792D 28D0                   <1>       sub    al, dl ; [PP_Counter] ; 4 - partition index
                                <1>
                                <1>       ; AL = Partition entry/index, 0 based
                                <1>       ;  0 -> hd 0, Partition Table offset = 0
                                <1>       ; 15 -> hd 3, Partition Table offset = 3
                                <1>
                                <1>       ;mov   [esi+LD_PartitionEntry], al
                                <1>
                                <1>       ; 15/07/2020
0000792F B404                   <1>       mov    ah, 4
                                <1>       ;sub   ah, [PP_Counter]
00007931 28D4                   <1>       sub    ah, dl
                                <1>
                                <1>       ; AH = Primary partition index, 0 to 3 ; pt entry
                                <1>       ;            (4 to 7 for logical disk partitions)
                                <1>
                                <1>       ;mov   [esi+LD_DParamEntry], ah
00007933 6689467C               <1>       mov    [esi+LD_PartitionEntry], ax
                                <1>
                                <1> loc_validate_next_hdp_partition1:
00007937 5A                     <1>       pop    edx ; **** ; dl = Physical drive number
00007938 5E                     <1>       pop    esi ; *** ; Masterboot partition table offset
                                <1>
                                <1> loc_validate_next_hdp_partition2:
                                <1>       ; ESI = PartitionTable offset
                                <1>       ; DL = Hard/Fixed disk drive number
                                <1>
                                <1>       ;dec   byte [hdc] ; 4 - partition index
                                <1>       ;jz    short pass_pt_this_hard_disk
                                <1>       ; 15/07/2020
00007939 FE0D[E38F0100]         <1>       dec    byte [PP_Counter] ; 4 - partition index
0000793F 7410                   <1>       jz     short pass_pt_this_hard_disk
                                <1>
00007941 83C610                 <1>       add    esi, 16 ; 10h
00007944 EB95                   <1>       jmp    short loc_validate_hdp_partition
                                <1>
                                <1> loc_not_any_extd_partitions:
                                <1>       ; 15/07/2020
00007946 C3                     <1>       retn
                                <1>
                                <1> loc_next_hd_partition_table:
00007947 FEC2                   <1>       inc    dl
                                <1>       ; 15/07/2020
                                <1>       ;add   esi, 32 ; next FDPT address
00007949 83C740                 <1>       add    edi, 64 ; next partition table destination
0000794C E95AFFFFFF             <1>         jmp     load_next_hd_partition_table
                                <1>
                                <1> pass_pt_this_hard_disk:
                                <1>       ;pop   esi ; FDPT (+ DPTE) address
                                <1>       ; 15/07/2020
00007951 5E                     <1>       pop    esi ; ** ; next FDPT (+ DPTE) address ptr
00007952 5F                     <1>       pop    edi ; * ; Ptable_hd?
                                <1>       ;pop   ecx
                                <1>       ;loop  loc_next_hd_partition_table
00007953 FE0D[0D6E0000]         <1>       dec    byte [hdc]
00007959 75EC                   <1>       jnz    short loc_next_hd_partition_table
                                <1>
                                <1>       ;cmp   byte [PP_Counter], 1
                                <1>       ;jnb   short load_extended_dos_partitions
                                <1>       ;; Empty partition table
                                <1>       ;retn
                                <1>
                                <1>       ; 11/08/2020
                                <1>       ; 17/07/2020
                                <1> check_extended_partitions:
                                <1>       ; 15/07/2020
                                <1>       ;cmp   byte [EP_Counter], 0
                                <1>       ;jna   short loc_not_any_extd_partitions
                                <1>       ; 13/08/2020
0000795B A0[E48F0100]           <1>       mov    al, [EP_Counter] ; 1st disk drv has extd partition
00007960 08C0                   <1>       or     al, al ; 0 ?
00007962 74E2                   <1>       jz     short loc_not_any_extd_partitions
                                <1>
                                <1> load_extended_dos_partitions:
                                <1>       ;mov   byte [hdc], 80h
                                <1>       ; 13/08/2020
00007964 A2[0D6E0000]           <1>       mov    byte [hdc], al ; 1st disk drv has extd partition
                                <1>       ; 25/08/2020
00007969 2C80                   <1>       sub    al, 80h
0000796B 740E                   <1>       jz     short loc_set_ext_ptable_hd0
0000796D C0E006                 <1>       shl    al, 6 ; * 64
00007970 0FB6F0                 <1>       movzx  esi, al
00007973 81C6[DE8E0100]         <1>       add    esi, PTable_hd0
00007979 EB05                   <1>       jmp    short next_hd_extd_partition
                                <1>
```

```
                                <1>       ; 25/08/2020
                                <1> loc_set_ext_ptable_hd0:
0000797B BE[DE8E0100]           <1>       mov   esi, PTable_hd0
                                <1>
                                <1> next_hd_extd_partition:
                                <1>       ; 17/07/2020
                                <1>       ;mov  byte [EP_Counter], 0 ; Reset for each physical disk
                                <1>       ; 13/08/2020
                                <1>       ;mov  byte [LD_Counter], 0 ; Reset logical drive index
00007980 66C705[E48F0100]00-    <1>       mov   word [EP_Counter], 0 ; Reset EP index and LD index
00007988 00                     <1>
                                <1>
00007989 56                     <1>       push  esi ; **** ; PTable_hd? offset
                                <1>
0000798A C605[E38F0100]04       <1>       mov   byte [PP_Counter], 4
                                <1>                          ; set for each extd partition table
                                <1>       ;;mov ecx, 4
                                <1>       ;mov  cl, 4
00007991 8A15[0D6E0000]         <1>       mov   dl, [hdc]
                                <1> hd_check_fs_id_05h:
00007997 8A4604                 <1>       mov   al, [esi+ptFileSystemID]
0000799A 3C05                   <1>       cmp   al, 05h ; Is it an extended dos partition ?
0000799C 7411                   <1>       je    short loc_set_ep_start_sector ; yes
                                <1> hd_check_fs_id_0Fh:
0000799E 3C0F                   <1>       cmp   al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
000079A0 740D                   <1>       je    short loc_set_ep_start_sector ; yes
                                <1>
                                <1> continue_to_check_ep:
                                <1>       ;add  esi, 16
                                <1>       ;loop hd_check_fs_id_05h
                                <1>       ; 15/07/2020
                                <1>       ;dec  cl
                                <1>       ;jz   short continue_check_ep_next_disk
000079A2 FE0D[E38F0100]         <1>       dec   byte [PP_Counter] ; 4 --> 0
000079A8 742D                   <1>       jz    short continue_check_ep_next_disk
000079AA 83C610                 <1>       add   esi, 16
000079AD EBE8                   <1>       jmp   short hd_check_fs_id_05h
                                <1>
                                <1> loc_set_ep_start_sector:
                                <1>       ; dl = [hdc] ; Drive number
                                <1>       ; 15/07/2020
000079AF 8B4E08                 <1>       mov   ecx, [esi+ptStartSector]
                                <1>       ; 30/08/2020
000079B2 890D[E68F0100]         <1>       mov   [MBR_EP_StartSector], ecx
                                <1>       ; 20/07/2020
                                <1> loc_validate_hde_partition_next:
000079B8 890D[EA8F0100]         <1>       mov   [EP_StartSector], ecx ; Extended partition's start sector
000079BE BB[DE8C0100]           <1>         mov ebx, MasterBootBuff
000079C3 803D[E28F0100]01       <1>       cmp   byte [HD_LBA_yes], 1 ; LBA ready = Yes
000079CA 7227                   <1>       jb    short loc_hd_load_ep_05h ; cf = 1 ; 20/07/2020
                                <1>       ; 11/08/2020
                                <1>       ; (BugFix for extended partition type 05h beyond CHS limit)
                                <1>       ; (Infact if extended partition starts at the beyond of CHS limit,
                                <1>       ;  it's partition ID must be 0Fh but they/somebodies had used 05h.)
                                <1>       ;cmp  al, 05h
                                <1>       ;je   short loc_hd_load_ep_05h
                                <1> loc_hd_load_ep_0Fh:
                                <1>       ; 04/01/2016
                                <1>       ;push ecx
                                <1>       ; 15/07/2020
                                <1>       ;mov  ecx, [esi+ptStartSector] ; sector number
                                <1>       ;mov  ebx, MasterBootBuff ; 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)
                                <1>       ;mov  ah, 1Bh ; LBA read
                                <1>       ;mov  al, 1 ; sector count
000079CC 66B8011B               <1>       mov   ax, 1B01h
000079D0 E85CD8FFFF             <1>       call  int13h
                                <1>       ;pop  ecx
                                <1>       ;jnc  short loc_hd_move_ep_table
                                <1>       ; 15/07/2020
000079D5 732E                   <1>       jnc   short loc_validate_hde_partition
                                <1>
                                <1> continue_check_ep_next_disk:
                                <1>       ; 15/07/2020
                                <1>       ;pop  edi ; PTable_ep?
000079D7 5E                     <1>       pop   esi ; **** ; PTable_hd?
000079D8 A0[B48A0100]           <1>       mov   al, [HF_NUM] ; number of hard disks
000079DD 047F                   <1>       add   al, 7Fh
000079DF 3805[0D6E0000]         <1>       cmp   [hdc], al
000079E5 730B                   <1>       jnb   short loc_validating_hd_partitions_ok
000079E7 83C640                 <1>       add   esi, 64
                                <1>       ; 15/07/2020
                                <1>       ;add  edi, 64
000079EA FE05[0D6E0000]         <1>       inc   byte [hdc]
000079F0 EB8E                   <1>       jmp   short next_hd_extd_partition
                                <1>
                                <1> loc_validating_hd_partitions_ok:
                                <1>       ; 15/07/2020
                                <1>       ;mov  al, [Last_DOS_DiskNo]
                                <1> loc_drv_init_retn:
000079F2 C3                     <1>       retn
                                <1>
                                <1> loc_hd_load_ep_05h:
                                <1>       ; 20/07/2020 ('diskio.s', int13h, cf = 1 -> bugfix)
                                <1>       ;clc    ; (Bug: int13h would not clear carry flag bit,
                                <1>       ;     ; even if there would not be an error)
                                <1>       ;     ; ((Fix: now, int13h procedure clears carry flag
                                <1>       ;     ;  at the entrance of it.. 20/07/2020))
                                <1>       ; 15/07/2020
                                <1>       ;push ecx
000079F3 8A7601                 <1>       mov   dh, [esi+ptBeginHead]
000079F6 668B4E02               <1>         mov     cx, [esi+ptBeginSector]
000079FA 66B80102               <1>       mov   ax, 0201h ; Read 1 sector
```

```
                        <1>       ;mov   ebx, MasterBootBuff
000079FE E82ED8FFFF      <1>       call   int13h ; 20/07/2020
                        <1>                      ; 'diskio.s' modification, 'clc'
                        <1>       ;pop   ecx
00007A03 72D2            <1>       jc     short continue_check_ep_next_disk
                        <1>       ; 15/07/2020
                        <1>       ;jmp   short loc_validate_hde_partition
                        <1>
                        <1>       ; 15/07/2020
                        <1> ;loc_hd_move_ep_table:
                        <1>       ;;pop  edi
                        <1>       ;;push edi  ; PTable_ep?
                        <1>       ;mov   edi, [esp]
                        <1>         ;mov esi, PartitionTable ; Extended
                        <1>       ;mov   ebx, esi
                        <1>       ;;mov  ecx, 16
                        <1>       ;mov   cl, 16
                        <1>              ;rep   movsd
                        <1>       ;mov   esi, ebx
                        <1> ;loc_set_hde_sub_partition_count:
                        <1>       ;mov   byte [PP_Counter], 4
                        <1>       ;mov   byte [EP_Counter], 0
                        <1>
                        <1> loc_validate_hde_partition:
                        <1>       ; 13/08/2020
                        <1>       ; 15/07/2020
                        <1>       ;mov   byte [PP_Counter], 4
00007A05 BE[9C8E0100]    <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
00007A0A 807E0400        <1>       cmp    byte [esi+ptFileSystemID], 0
00007A0E 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
00007A10 FE05[E48F0100]  <1>       inc    byte [EP_Counter] ; current (sub partition) index
                        <1>                               ; in current extended partition
                        <1>
00007A16 BF[EA8F0100]    <1>       mov    edi, EP_StartSector
                        <1>
                        <1>       ; Input -> ESI = PartitionTable offset
                        <1>       ; DL = Hard disk drive number
                        <1>       ; EDI = Extended partition start sector pointer
00007A1B E869000000      <1>       call   validate_hd_fat_partition
                        <1>       ;pop   ecx ; *
00007A20 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
00007A22 8A15[0D6E0000]  <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
00007A28 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
```

```
00007A2A A0[0D6E0000]        <1>       mov    al, [hdc] ; Hard disk drive number (>=80h)
00007A2F 88C2                <1>       mov    dl, al ; mov dl, [hdc]
00007A31 2C7F                <1>       sub    al, 7Fh
                             <1>                  ; 1 to 4
00007A33 C0E002              <1>       shl    al, 2 ; 4 to 16
00007A36 2A05[E38F0100]      <1>       sub    al, [PP_Counter] ; al - (4 - partition index)
                             <1>                     ; (disk number * 4) + partition index
                             <1>
                             <1>       ; AL = Partition entry/index, 0 based
                             <1>         ;  0 -> hd 0, Partition Table offset = 0
                             <1>         ; 15 -> hd 3, Partition Table offset = 3
                             <1>
                             <1>       ;mov   ah, 4 ; Logical dos partition (>= 4)
                             <1>       ;add   ah, [EP_Counter]
                             <1>              ; Logical disk partition index = 4 to 7
                             <1>              ; (Primary disk partition index = 0 to 3)
                             <1>
                             <1>       ; 13/08/2020
00007A3C 8A25[E58F0100]      <1>       mov    ah, [LD_Counter] ; Logical drive index number
                             <1>                              ; (in current extended partition)
00007A42 80C404              <1>       add    ah, 4 ; 4 to 7
                             <1>
                             <1>       ; 15/07/2020
                             <1>       ; CX -> AX
                             <1>       ;; 06/01/2016 (TRDOS v2.0)
                             <1>       ;; BUGFIX *
                             <1>       ;;mov  [esi+LD_PartitionEntry], cl
                             <1>       ;;mov  [esi+LD_DParamEntry], ch
                             <1>       ;mov   [esi+LD_PartitionEntry], cx
00007A45 6689467C            <1>       mov    [esi+LD_PartitionEntry], ax
                             <1>
00007A49 8A0D[A5400100]      <1>       mov    cl, [Last_DOS_DiskNo]
00007A4F 80C141              <1>       add    cl, 'A'
00007A52 880E                <1>       mov    [esi+LD_Name], cl
                             <1>
                             <1>       ; 17/07/2020
                             <1>       ;cmp   cl, 'Z'
                             <1>       ;jb    short logical_drive_count_ok_for_next
                             <1>       ;pop   esi ; ***
                             <1>       ;pop   esi ; ****
                             <1>       ;retn
                             <1>
                             <1> ;logical_drive_count_ok_for_next:
                             <1>
                             <1>       ;; 15/07/2020
                             <1>       ;inc   byte [EP_Counter]
                             <1>       ; 13/08/2020
00007A54 FE05[E58F0100]      <1>       inc    byte [LD_Counter]
                             <1>
                             <1>       ;mov   dl, [hdc]
                             <1>
                             <1>       ; 17/07/2020
                             <1>       ;; Now,
                             <1>       ;; we are swimming in deep of an extended partition !!!
                             <1>       ; (! sub or chained extended partition tables !)
                             <1>       ; ((Logical dos partitions in extended partition were called
                             <1>       ;  as 'mini disk partition' in msdos 6.0 source code.))
                             <1>
                             <1> validate_next_minidisk_partition:
                             <1>       ; 13/08/2020
                             <1>       ;pop   esi ; *** ; Extended partition table offset
                             <1>
                             <1>       ; 17/07/2020
                             <1>       ;cmp   byte [EP_Counter], 4
                             <1>       ; 13/08/2020
00007A5A 803D[E58F0100]04    <1>       cmp    byte [LD_Counter], 4 ; maximum 4 logical disks
                             <1>                                   ; per extended partition
00007A61 0F8370FFFFFF        <1>       jnb    continue_check_ep_next_disk
                             <1>
                             <1> validate_next_minidisk_partition_ok:
                             <1>       ; 13/08/2020
                             <1>       ;dec   byte [PP_Counter] ; 4 --> 0
                             <1>       ;jz    continue_check_ep_next_disk
                             <1>
                             <1>       ;cmp   esi, PartitionTable+64
                             <1>       ;jnb   continue_check_ep_next_disk
                             <1>
                             <1>       ;add   esi, 16
                             <1>       ; 13/08/2020
00007A67 BE[AC8E0100]        <1>       mov    esi, PartitionTable+16
                             <1>
                             <1>       ; 20/07/2020
00007A6C 8A4604              <1>       mov    al, [esi+ptFileSystemID]
                             <1>
                             <1>       ; 20/07/2020
00007A6F 3C05                <1>       cmp    al, 05h ; Is it an extended dos partition ?
00007A71 7408                <1>       je     short loc_minidisk_next_ep_lba_chs ; 17/07/2020
00007A73 3C0F                <1>       cmp    al, 0Fh ; Is it an extended win4 (LBA mode) partition ?
00007A75 0F855CFFFFFF        <1>       jne    continue_check_ep_next_disk ; AL must be 0 here
                             <1>                                  ; (when it is not 05h or 0Fh)
                             <1>                                  ; If AL is not ZERO -> EP Bug!
                             <1>                                  ; (!Microsoft DOS convention!)
                             <1> loc_minidisk_next_ep_lba_chs:
                             <1>       ; 17/07/2020
00007A7B 8B4E08              <1>       mov    ecx, [esi+ptStartSector] ; relative start sector number
                             <1>       ;add   ecx, [EP_StartSector]
                             <1>       ; 30/08/2020
00007A7E 030D[E68F0100]      <1>       add    ecx, [MBR_EP_StartSector]
                             <1>       ; 20/07/2020
00007A84 E92FFFFFFF          <1>       jmp    loc_validate_hde_partition_next
                             <1>
                             <1> validate_hd_fat_partition:
                             <1>       ; 17/07/2020
                             <1>       ; 15/07/2020
                             <1>       ;      (optimization)
```

```
                        <1>       ; 14/07/2020
                        <1>       ;      (fat16 -big- partition search bugfix)
                        <1>       ; 27/12/2017
                        <1>       ; 12/02/2016
                        <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 07/08/2011
                        <1>       ; 23/07/2011
                        <1>       ; Input
                        <1>       ;   DL = Hard/Fixed Disk Drive Number
                        <1>       ;   ESI = PartitionTable offset
                        <1>       ;   EDI = Extend. Part. Start Sector Pointer
                        <1>       ;   EDI = 0 -> Primary Partition
                        <1>       ;   byte [Last_DOS_DiskNo]
                        <1>       ; Output
                        <1>       ;  cf=0 -> Validated
                        <1>       ;   ESI = Logical dos drv desc. table
                        <1>       ;   EBX = FAT boot sector buffer
                        <1>       ;   byte [Last_DOS_DiskNo]
                        <1>       ;  cf=1 -> Not a valid FAT partition
                        <1>       ; EAX, EDX, ECX, EDI -> changed
                        <1>
                        <1>       ;mov   esi, PartitionTable
00007A89 8A6604         <1>       mov    ah, [esi+ptFileSystemID]
00007A8C B002           <1>       mov    al, 2 ; 27/12/2017
00007A8E 80FC06         <1>       cmp    ah, 06h ; FAT16 CHS partition (>=32MB)
                        <1>       ; 12/02/2016
                        <1>       ;;jb   short loc_not_a_valid_fat_partition2
                        <1>       ;jnb   short vhdp_FAT16_32
                        <1>       ; 14/07/2020 (BugFix)
00007A91 7711           <1>       ja     short vhdp_FAT16_32
00007A93 7425           <1>       je     short loc_set_valid_hd_partition_params
                        <1>
                        <1> vhdp_FAT12_16:
                        <1>       ; 27/12/2017
00007A95 FEC8           <1>       dec    al ; mov al, 1
00007A97 38C4           <1>       cmp    ah, al ; 1 ; FAT12 partition
00007A99 741F           <1>       je     short loc_set_valid_hd_partition_params
                        <1>       ;
00007A9B FEC0           <1>       inc    al ; mov al, 2
00007A9D 80FC04         <1>       cmp    ah, 04h ; FAT16 CHS partition (< 32MB)
00007AA0 7418           <1>       je     short loc_set_valid_hd_partition_params
                        <1>
                        <1>       ; 15/07/2020
                        <1>       ; (ah = 05h, 02h or 03h)
                        <1> loc_not_a_valid_fat_partition1:
00007AA2 F9             <1>       stc
                        <1>       ; cf=1
00007AA3 C3             <1>       retn
                        <1>
                        <1> vhdp_FAT16_32:
                        <1>       ; 15/07/2020
                        <1>       ;mov   al, 3
00007AA4 FEC0           <1>       inc    al
00007AA6 80FC0C         <1>       cmp    ah, 0Ch ; FAT32 LBA partition
00007AA9 740F           <1>       je     short loc_set_valid_hd_partition_params
00007AAB 7706           <1>       ja     short vhdp_check_FAT16_lba
                        <1>
                        <1> vhdp_check_FAT32_chs:
00007AAD 80FC0B         <1>       cmp    ah, 0Bh ; FAT32 CHS partition
00007AB0 7408           <1>       je     short loc_set_valid_hd_partition_params
                        <1>       ;jne   short loc_not_a_valid_fat_partition1
                        <1>
                        <1>       ;stc
                        <1> loc_not_a_valid_fat_partition2:
00007AB2 C3             <1>       retn
                        <1>
                        <1> vhdp_check_FAT16_lba:
00007AB3 80FC0E         <1>       cmp    ah, 0Eh ; FAT16 LBA partition
00007AB6 75EA           <1>       jne    short loc_not_a_valid_fat_partition1
                        <1>
                        <1>       ;mov   al, 2
00007AB8 FEC8           <1>       dec    al
                        <1>
                        <1> loc_set_valid_hd_partition_params:
                        <1>       ; 15/07/2020
                        <1>       ;inc   byte [Last_DOS_DiskNo] ; > 1
                        <1>       ;
00007ABA 31DB           <1>       xor    ebx, ebx
00007ABC 8A3D[A5400100] <1>       mov    bh, [Last_DOS_DiskNo] ; * 256
00007AC2 FEC7           <1>       inc    bh ; 15/07/2020
00007AC4 81C300010900   <1>       add    ebx, Logical_DOSDisks
                        <1>       ;
00007ACA C6430102       <1>       mov    byte [ebx+LD_DiskType], 2
00007ACE 885302         <1>       mov    byte [ebx+LD_PhyDrvNo], dl
                        <1>       ;mov   byte [ebx+LD_FATType], al ; 2 or 3
                        <1>       ;mov   byte [ebx+LD_FSType], ah ; 06h, 0Eh, 0Bh, 0Ch
00007AD1 66894303       <1>       mov    word [ebx+LD_FATType], ax
                        <1>       ;
00007AD5 8B4E08         <1>       mov    ecx, [esi+ptStartSector]
00007AD8 09FF           <1>       or     edi, edi
00007ADA 7402           <1>       jz     short pass_hd_FAT_ep_start_sector_adding
                        <1> loc_add_hd_FAT_ep_start_sector:
                        <1>       ; 17/07/2020
00007ADC 030F           <1>       add    ecx, [edi]
                        <1> pass_hd_FAT_ep_start_sector_adding:
00007ADE 894B6C         <1>       mov    [ebx+LD_StartSector], ecx
                        <1> loc_hd_FAT_logical_drv_init:
00007AE1 89DD           <1>       mov    ebp, ebx
                        <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
00007AE3 A0[E28F0100]   <1>       mov    al, [HD_LBA_yes] ; 07/01/2016
00007AE8 884305         <1>       mov    [ebx+LD_LBAYes], al
00007AEB BB[EE8F0100]   <1>       mov    ebx, DOSBootSectorBuff ; buffer address
00007AF0 08C0           <1>       or     al, al
00007AF2 740C           <1>       jz     short loc_hd_FAT_drv_init_load_bs_chs
                        <1> loc_hd_FAT_drv_init_load_bs_lba:
```

```
                                <1>       ; DL = Physical drive number
                                <1>       ;mov   ecx, [esi+ptStartSector] ; sector number
                                <1>       ;mov   ebx, DOSBootSectorBuff ; buffer address
                                <1>       ; LBA read/write (with private LBA function)
                                <1>       ;((Retro UNIX 386 v1 - DISK I/O code by Erdogan Tan))
                                <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
00007AF4 B41B                   <1>       mov    ah, 1Bh ; LBA read
00007AF6 B001                   <1>       mov    al, 1 ; sector count
00007AF8 E834D7FFFF             <1>       call   int13h
00007AFD 7313                   <1>       jnc    short loc_hd_drv_FAT_boot_validation
                                <1> loc_not_a_valid_fat_partition3:
00007AFF C3                     <1>       retn
                                <1> loc_hd_FAT_drv_init_load_bs_chs:
00007B00 8A7601                 <1>       mov    dh, [esi+ptBeginHead]
00007B03 668B4E02               <1>       mov    cx, [esi+ptBeginSector]
00007B07 66B80102               <1>       mov    ax, 0201h ; Read 1 sector
                                <1>       ;mov   ebx, DOSBootSectorBuff
00007B0B E821D7FFFF             <1>       call   int13h
00007B10 72ED                   <1>       jc     short loc_not_a_valid_fat_partition3
                                <1> loc_hd_drv_FAT_boot_validation:
                                <1>       ;mov   esi, DOSBootSectorBuff
00007B12 89DE                   <1>       mov    esi, ebx
00007B14 6681BEFE01000055AA     <1>       cmp    word [esi+BS_Validation], 0AA55h
00007B1D 7512                   <1>       jne    short loc_not_a_valid_fat_partition4
00007B1F 807E15F8               <1>       cmp    byte [esi+BPB_Media], 0F8h
00007B23 750C                   <1>       jne    short loc_not_a_valid_fat_partition4
                                <1>
                                <1>       ; 27/12/2017
00007B25 807D0303               <1>       cmp    byte [ebp+LD_FATType], 3
00007B29 7508                   <1>       jne    short loc_hd_FAT16_BPB
                                <1>
                                <1> loc_hd_drv_FAT32_boot_validation:
00007B2B 807E4229               <1>       cmp    byte [esi+BS_FAT32_BootSig], 29h
00007B2F 7416                   <1>       je     short loc_hd_FAT32_BPB
                                <1>
                                <1> loc_not_a_valid_fat_partition4:
00007B31 F9                     <1>       stc
00007B32 C3                     <1>       retn
                                <1>
                                <1> loc_hd_FAT16_BPB:
00007B33 807E2629               <1>       cmp    byte [esi+BS_BootSig], 29h
00007B37 75F8                   <1>       jne    short loc_not_a_valid_fat_partition4
                                <1>
00007B39 66837E1600             <1>       cmp    word [esi+BPB_FATSz16], 0
00007B3E 7607                   <1>       jna    short loc_hd_big_FAT16_BPB
00007B40 B920000000             <1>       mov    ecx, 32
00007B45 EB05                   <1>       jmp    short loc_hd_move_FAT_BPB
                                <1>
                                <1> loc_hd_FAT32_BPB:
                                <1>       ;cmp   word [esi+BPB_FATSz16], 0
                                <1>       ;ja    short loc_not_a_valid_fat_partition4
                                <1> loc_hd_big_FAT16_BPB:
00007B47 B92D000000             <1>       mov    ecx, 45
                                <1>
                                <1> loc_hd_move_FAT_BPB:
00007B4C 89EF                   <1>       mov    edi, ebp
                                <1>       ;mov   esi, ebx ; Boot sector
00007B4E 57                     <1>       push   edi
00007B4F 83C706                 <1>       add    edi, LD_BPB
00007B52 F366A5                 <1>       rep    movsw
00007B55 5E                     <1>       pop    esi
00007B56 0FB74614               <1>       movzx  eax, word [esi+LD_BPB+BPB_RsvdSecCnt]
00007B5A 03466C                 <1>       add    eax, [esi+LD_StartSector]
00007B5D 894660                 <1>       mov    [esi+LD_FATBegin], eax
00007B60 807E0303               <1>       cmp    byte [esi+LD_FATType], 3
00007B64 7224                   <1>       jb     short loc_set_FAT16_RootDirLoc
                                <1> loc_set_FAT32_RootDirLoc:
00007B66 8B462A                 <1>       mov    eax, [esi+LD_BPB+BPB_FATSz32]
00007B69 0FB65E16               <1>         movzx      ebx, byte [esi+LD_BPB+BPB_NumFATs]
00007B6D F7E3                   <1>       mul    ebx
00007B6F 034660                 <1>       add    eax, [esi+LD_FATBegin]
                                <1> loc_set_FAT32_data_begin:
00007B72 894668                 <1>       mov    [esi+LD_DATABegin], eax
00007B75 894664                 <1>       mov    [esi+LD_ROOTBegin], eax
                                <1>       ; If Root Directory Cluster <> 2 then
                                <1>       ; change the beginning sector value
                                <1>       ; of the root dir by adding sector offset.
00007B78 8B4632                 <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
00007B7B 83E802                 <1>       sub    eax, 2
00007B7E 7435                   <1>       jz     short short loc_set_32bit_FAT_total_sectors
                                <1>       ;movzx ebx, byte [esi+LD_BPB+BPB_SecPerClust]
00007B80 8A5E13                 <1>       mov    bl, [esi+LD_BPB+BPB_SecPerClust]
00007B83 F7E3                   <1>       mul    ebx
00007B85 014664                 <1>       add    [esi+LD_ROOTBegin], eax
00007B88 EB2B                   <1>       jmp    short loc_set_32bit_FAT_total_sectors
                                <1>       ;
                                <1> loc_set_FAT16_RootDirLoc:
00007B8A 0FB64616               <1>       movzx  eax, byte [esi+LD_BPB+BPB_NumFATs]
00007B8E 0FB7561C               <1>       movzx  edx, word [esi+LD_BPB+BPB_FATSz16]
00007B92 F7E2                   <1>       mul    edx
00007B94 034660                 <1>       add    eax, [esi+LD_FATBegin]
00007B97 894664                 <1>       mov    [esi+LD_ROOTBegin], eax
                                <1> loc_set_FAT16_data_begin:
00007B9A 894668                 <1>       mov    [esi+LD_DATABegin], eax
                                <1>       ;mov   eax, 20h  ; Size of a directory entry
                                <1>       ;;movzx      edx, word [esi+LD_BPB+BPB_RootEntCnt]
                                <1>         ;mov     dx, [esi+LD_BPB+BPB_RootEntCnt]
                                <1>         ;mul edx
                                <1>       ;;mov  ecx, 511
                                <1>       ;mov   cx, 511
                                <1>       ;add   eax, ecx
                                <1>       ;inc   ecx ; 512
                                <1>       ;div   ecx
                                <1>       ; 14/07/2020
00007B9D 0FB74617               <1>       movzx  eax, word [esi+LD_BPB+BPB_RootEntCnt]
```

```
00007BA1 6683C00F            <1>       add    ax, 15
00007BA5 66C1E804            <1>       shr    ax, 4 ; / 16 ; (16 entries per sector)
00007BA9 014668              <1>       add    [esi+LD_DATABegin], eax
                             <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
00007BAC 668B4619            <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
00007BB0 6685C0              <1>       test   ax, ax
                             <1>       ;jz    short loc_set_32bit_FAT_total_sectors
                             <1> ;loc_set_16bit_FAT_total_sectors:
                             <1>       ;mov   [esi+LD_TotalSectors], eax
                             <1>       ;jmp   short loc_set_hd_FAT_cluster_count
                             <1>       ; 14/07/2020
00007BB3 7503                <1>       jnz    short loc_set_hd_FAT_cluster_count
                             <1> loc_set_32bit_FAT_total_sectors:
00007BB5 8B4626              <1>       mov    eax, [esi+LD_BPB+BPB_TotalSec32]
                             <1>       ;mov   [esi+LD_TotalSectors], eax
                             <1> loc_set_hd_FAT_cluster_count:
00007BB8 894670              <1>       mov    [esi+LD_TotalSectors], eax ; 14/07/2020
00007BBB 8B5668              <1>       mov    edx, [esi+LD_DATABegin]
00007BBE 2B566C              <1>       sub    edx, [esi+LD_StartSector]
00007BC1 29D0                <1>       sub    eax, edx
00007BC3 31D2                <1>       xor    edx, edx ; 0
00007BC5 0FB64E13            <1>         movzx   ecx, byte [esi+LD_BPB+BPB_SecPerClust]
00007BC9 F7F1                <1>         div  ecx
00007BCB 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
00007BCE E855010000          <1>       call   get_free_FAT_sectors
00007BD3 720D                <1>       jc     short loc_validate_hd_FAT_partition_retn
00007BD5 894674              <1>       mov    [esi+LD_FreeSectors], eax
00007BD8 C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6  ; Volume Name Reset
                             <1>
                             <1>       ; 15/07/2020
00007BDC FE05[A5400100]      <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:
00007BE2 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
00007BE3 8A6604              <1>       mov    ah, [esi+ptFileSystemID]
00007BE6 80FCA1              <1>       cmp    ah, 0A1h ; SINGLIX FS1 (trfs1) partition
00007BE9 7549                <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                             <1> loc_set_valid_hd_fs_partition_params:
00007BEB FE05[A5400100]      <1>       inc    byte [Last_DOS_DiskNo] ; > 1
00007BF1 30C0                <1>       xor    al, al ; mov al, 0
                             <1>       ;mov   [drv], dl
00007BF3 29DB                <1>       sub    ebx, ebx ; 0
00007BF5 8A3D[A5400100]      <1>       mov    bh, [Last_DOS_DiskNo]
00007BFB 81C300010900        <1>       add    ebx, Logical_DOSDisks
00007C01 C6430102            <1>       mov    byte [ebx+LD_DiskType], 2
00007C05 885302              <1>       mov    [ebx+LD_PhyDrvNo], dl
                             <1>       ;mov   [ebx+LD_FATType], al ; 0
                             <1>       ;mov   [ebx+LD_FSType], ah
00007C08 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:
00007C0C 89DD                <1>       mov    ebp, ebx ; 10/01/2016
                             <1>       ;mov   dl, [ebx+LD_PhyDrvNo]
00007C0E A0[E28F0100]        <1>       mov    al, [HD_LBA_yes] ; 10/01/2016
00007C13 884305              <1>       mov    [ebx+LD_LBAYes], al
00007C16 89DE                <1>       mov    esi, ebx
00007C18 BB[EE8F0100]        <1>       mov    ebx, DOSBootSectorBuff ; buffer address
00007C1D 08C0                <1>       or     al, al
00007C1F 7515                <1>       jnz    short loc_hd_fs_drv_init_load_bs_lba
                             <1> loc_hd_fs_drv_init_load_bs_chs:
00007C21 8A7601              <1>       mov    dh, [esi+ptBeginHead]
00007C24 668B4E02            <1>       mov    cx, [esi+ptBeginSector]
00007C28 66B80102            <1>       mov    ax, 0201h ; Read 1 sector
                             <1>       ;mov   ebx, DOSBootSectorBuff
00007C2C E800D6FFFF          <1>       call   int13h
00007C31 7311                <1>       jnc    short loc_hd_drv_fs_boot_validation
                             <1> loc_validate_hd_fs_partition_err_retn:
00007C33 C3                  <1>       retn
                             <1> loc_validate_hd_fs_partition_stc_retn:
00007C34 F9                  <1>       stc
00007C35 C3                  <1>       retn
                             <1> loc_hd_fs_drv_init_load_bs_lba:
                             <1>       ; DL = Physical drive number
                             <1>       ;mov   esi, ebx
00007C36 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)
00007C39 B41B                   <1>       mov    ah, 1Bh ; LBA read
00007C3B B001                   <1>       mov    al, 1 ; sector count
00007C3D E8EFD5FFFF             <1>       call   int13h
00007C42 72EF                   <1>       jc     short loc_validate_hd_fs_partition_err_retn
                                <1> loc_hd_drv_fs_boot_validation:
                                <1>       ;mov   esi, DOSBootSectorBuff
00007C44 89DE                   <1>       mov    esi, ebx ; Boot sector buffer
00007C46 6681BEFE01000055AA     <1>       cmp    word [esi+BS_Validation], 0AA55h
00007C4F 75E3                   <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                                <1>         ;
                                <1>       ;Singlix FS Extensions to TR-DOS (7/6/2009)
00007C51 66817E034653           <1>       cmp    word [esi+bs_FS_Identifier], 'FS' ; 03/02/2018
00007C57 75DB                   <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                                <1>         ;'A1h' check is not necessary
                                <1>       ;  if 'FS' check is passed as OK/Yes.
00007C59 807E09A1               <1>       cmp    byte [esi+bs_FS_PartitionID], 0A1h
00007C5D 75D5                   <1>       jne    short loc_validate_hd_fs_partition_stc_retn
                                <1>       ;
00007C5F 89EF                   <1>       mov    edi, ebp ; 10/01/2016
                                <1>       ;
00007C61 8A462D                 <1>       mov    al, byte [esi+bs_FS_LBA_Ready]
00007C64 884705                 <1>       mov    [edi+LD_FS_LBAYes], al
                                <1>       ;
                                <1>       ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
00007C67 8A4608                 <1>       mov    al, [esi+bs_FS_MediaAttrib]
00007C6A 884706                 <1>       mov    byte [edi+LD_FS_MediaAttrib], al
                                <1>       ;
00007C6D 8A460A                 <1>       mov    al, [esi+bs_FS_VersionMaj]
00007C70 884707                 <1>       mov    [edi+LD_FS_VersionMajor], al
                                <1>       ;
00007C73 668B4606               <1>       mov    ax, [esi+bs_FS_BytesPerSec]
00007C77 66894711               <1>       mov    [edi+LD_FS_BytesPerSec], ax
00007C7B 8A462E                 <1>       mov    al, [esi+bs_FS_SecPerTrack]
00007C7E 30E4                   <1>       xor    ah, ah ; 09/12/2017
00007C80 6689471E               <1>       mov    [edi+LD_FS_SecPerTrack], ax
00007C84 8A462F                 <1>       mov    al, [esi+bs_FS_Heads]
00007C87 66894720               <1>       mov    [edi+LD_FS_NumHeads], ax
                                <1>       ;
00007C8B 8B4628                 <1>       mov    eax, [esi+bs_FS_UnDelDirD]
00007C8E 894722                 <1>       mov    [edi+LD_FS_UnDelDirD], eax
00007C91 8B5618                 <1>       mov    edx, [esi+bs_FS_MATLocation]
00007C94 89570C                 <1>       mov    [edi+LD_FS_MATLocation], edx
00007C97 8B461C                 <1>       mov    eax, [esi+bs_FS_RootDirD]
00007C9A 894708                 <1>       mov    [edi+LD_FS_RootDirD], eax
00007C9D 8B460C                 <1>       mov    eax, [esi+bs_FS_BeginSector]
00007CA0 89476C                 <1>       mov    [edi+LD_FS_BeginSector], eax
00007CA3 8B4710                 <1>       mov    eax, [edi+bs_FS_VolumeSize]
00007CA6 894770                 <1>       mov    [edi+LD_FS_VolumeSize], eax
                                <1>       ;
00007CA9 89D0                   <1>       mov    eax, edx ; [edi+LD_FS_MATLocation]
00007CAB 03476C                 <1>       add    eax, [edi+LD_FS_BeginSector]
00007CAE 89FE                   <1>       mov    esi, edi
                                <1> mread_hd_fs_MAT_sector:
                                <1>         ;mov  ebx, DOSBootSectorBuff
00007CB0 B901000000             <1>       mov    ecx, 1
00007CB5 E8E6AE0000             <1>       call   disk_read
00007CBA 7248                   <1>       jc     short loc_validate_hd_fs_partition_retn
                                <1>       ; EDI will not be changed
00007CBC 89DE                   <1>       mov    esi, ebx
                                <1> use_hdfs_mat_sector_params:
00007CBE 8B460C                 <1>       mov    eax, [esi+FS_MAT_DATLocation]
00007CC1 894714                 <1>       mov    [edi+LD_FS_DATLocation], eax
00007CC4 8B4610                 <1>       mov    eax, [esi+FS_MAT_DATScount]
00007CC7 894718                 <1>       mov    [edi+LD_FS_DATSectors], eax
00007CCA 8B4614                 <1>       mov    eax, [esi+FS_MAT_FreeSectors]
00007CCD 894774                 <1>         mov     [edi+LD_FS_FreeSectors], eax
00007CD0 8B4618                 <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
00007CD3 894778                 <1>       mov    [edi+LD_FS_FirstFreeSector], eax
00007CD6 8B4708                 <1>       mov    eax, [edi+LD_FS_RootDirD]
00007CD9 03476C                 <1>       add    eax, [edi+LD_FS_BeginSector]
00007CDC 89FE                   <1>       mov    esi, edi
                                <1> read_hd_fs_RDT_sector:
00007CDE BB[EE8F0100]           <1>       mov    ebx, DOSBootSectorBuff
                                <1>       ;mov   ecx, 1
00007CE3 B101                   <1>       mov    cl, 1
00007CE5 E8B6AE0000             <1>       call   disk_read
00007CEA 7218                   <1>       jc     short loc_validate_hd_fs_partition_retn
                                <1>       ; EDI will not be changed
00007CEC 89DE                   <1>       mov    esi, ebx
                                <1> use_hdfs_RDT_sector_params:
00007CEE 8B461C                 <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
00007CF1 894728                 <1>       mov    [edi+LD_FS_VolumeSerial], eax
00007CF4 57                     <1>       push   edi
                                <1>       ;mov   ecx, 16
00007CF5 B110                   <1>       mov    cl, 16
00007CF7 83C640                 <1>       add    esi, FS_RDT_VolumeName
00007CFA 83C72C                 <1>       add    edi, LD_FS_VolumeName
00007CFD F3A5                   <1>       rep    movsd ; 64 bytes
00007CFF 5E                     <1>       pop    esi
                                <1>              ; Volume Name Reset
00007D00 C6467E06               <1>         mov     byte [esi+LD_FS_MediaChanged], 6
                                <1>       ;
                                <1>         ;mov cl, [Last_DOS_DiskNo]
                                <1>       ;add   cl, 'A'
                                <1>       ;mov   [esi+LD_FS_Name], cl
                                <1>
                                <1> loc_validate_hd_fs_partition_retn:
00007D04 C3                     <1>       retn
                                <1>
                                <1> load_masterboot:
                                <1>       ; 14/07/2020 (Reset function has been removed)
```

```
                                <1>       ;
                                <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 2005 - 2011
                                <1>       ; input -> DL = drive number
                                <1> ;     mov   ah, 0Dh ; Alternate disk reset
                                <1> ;     call  int13h
                                <1> ;     jnc   short pass_reset_error
                                <1> ;harddisk_error:
                                <1> ;     retn
                                <1> ;pass_reset_error:
00007D05 BB[DE8C0100]           <1>       mov   ebx, MasterBootBuff
00007D0A 66B80102               <1>       mov   ax, 0201h
00007D0E 66B90100               <1>       mov   cx, 1
00007D12 30F6                   <1>       xor   dh, dh
00007D14 E818D5FFFF             <1>       call  int13h
00007D19 720C                   <1>       jc    short harddisk_error
                                <1>       ;
00007D1B 66813D[DC8E0100]55-    <1>       cmp   word [MBIDCode], 0AA55h
00007D23 AA                     <1>
00007D24 7401                   <1>       je    short load_masterboot_ok
00007D26 F9                     <1>       stc
                                <1> harddisk_error:
                                <1> load_masterboot_ok:
00007D27 C3                     <1>       retn
                                <1>
                                <1> get_free_FAT_sectors:
                                <1>       ; 21/12/2017
                                <1>       ; 29/02/2016
                                <1>       ; 13/02/2016
                                <1>       ; 04/02/2016
                                <1>       ; 07/01/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 11/07/2010
                                <1>       ; 21/06/2009
                                <1>       ; INPUT: ESI = Logical DOS Drive Description Table address
                                <1>       ; OUTPUT: STC => Error
                                <1>         ;   cf = 0 and EAX = Free FAT sectors
                                <1>       ; Also, related parameters and FAT buffer will be reset and updated
                                <1>
00007D28 31C0                   <1>       xor   eax, eax
                                <1>       ;mov  [esi+LD_FreeSectors], eax ; Reset
                                <1>
00007D2A 807E0302               <1>         cmp     byte [esi+LD_FATType], 2
00007D2E 7654                   <1>       jna   short loc_gfc_get_fat_free_clusters
                                <1>
                                <1>       ; 29/02/2016
00007D30 48                     <1>       dec   eax ; 0FFFFFFFFh
00007D31 89463A                 <1>       mov   [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count (reset)
00007D34 89463E                 <1>       mov   [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster (reset)
00007D37 40                     <1>       inc   eax ; 0
                                <1>       ;
00007D38 668B4636               <1>       mov   ax, [esi+LD_BPB+BPB_FSInfo]
00007D3C 03466C                 <1>       add   eax, [esi+LD_StartSector]
                                <1>
00007D3F BB[EE8F0100]           <1>       mov   ebx, DOSBootSectorBuff
00007D44 B901000000             <1>       mov   ecx, 1
00007D49 E852AE0000             <1>       call  disk_read
00007D4E 7301                   <1>       jnc   short loc_gfc_check_fsinfo_signs
                                <1> retn_gfc_get_fsinfo_sec:
00007D50 C3                     <1>       retn
                                <1>
                                <1> loc_gfc_check_fsinfo_signs:
00007D51 BB[EE8F0100]           <1>       mov   ebx, DOSBootSectorBuff ; 13/02/2016
00007D56 813B52526141           <1>         cmp     dword [ebx], 41615252h
00007D5C 7524                   <1>       jne   short retn_gfc_get_fsinfo_stc
                                <1>       ;add  ebx, 484
                                <1>       ;cmp  dword [ebx], 61417272h
00007D5E 81BBE4010000727241-    <1>       cmp   dword [ebx+484], 61417272h
00007D67 61                     <1>
00007D68 7518                   <1>       jne   short retn_gfc_get_fsinfo_stc
                                <1>       ;add  ebx, 4
                                <1>       ;mov  eax, [ebx]
00007D6A 8B83E8010000           <1>       mov   eax, [ebx+488]
                                <1>       ; 29/02/2016
00007D70 89463A                 <1>       mov   [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
00007D73 8B93EC010000           <1>       mov   edx,  [ebx+492]
00007D79 89463E                 <1>       mov   [esi+LD_BPB+BPB_Reserved+4], eax ; First Free Cluster
                                <1>       ; 21/12/2017
00007D7C 89C3                   <1>       mov   ebx, eax ; (initial value = 0FFFFFFFFh)
00007D7E 43                     <1>       inc   ebx ; 0FFFFFFFFh -> 0
00007D7F 7513                   <1>       jnz   short short retn_from_get_free_fat32_clusters
00007D81 C3                     <1>       retn
                                <1>
                                <1> retn_gfc_get_fsinfo_stc:
00007D82 F9                     <1>       stc
00007D83 C3                     <1>       retn
                                <1>
                                <1> loc_gfc_get_fat_free_clusters:
                                <1>       ;mov  eax, 2
00007D84 B002                   <1>       mov   al, 2
                                <1>       ;mov  [FAT_CurrentCluster], eax
                                <1> loc_gfc_loop_get_next_cluster:
00007D86 E8EB4F0000             <1>       call  get_next_cluster
00007D8B 730E                   <1>       jnc   short loc_gfc_free_fat_clusters_cont
00007D8D 21C0                   <1>       and   eax, eax
00007D8F 7411                   <1>       jz    short loc_gfc_pass_inc_free_cluster_count
                                <1>
                                <1> retn_from_get_free_fat_clusters:
00007D91 8B4674                 <1>       mov   eax, [esi+LD_FreeSectors] ; Free clusters !
                                <1> retn_from_get_free_fat32_clusters:
00007D94 0FB65E13               <1>         movzx     ebx, byte [esi+LD_BPB+BPB_SecPerClust]
00007D98 F7E3                   <1>             mul   ebx
                                <1>       ;mov  [esi+LD_FreeSectors], eax ; Free sectors
                                <1> retn_get_free_sectors_calc:
00007D9A C3                     <1>       retn
                                <1>
```

```
                        <1> loc_gfc_free_fat_clusters_cont:
00007D9B 09C0           <1>         or      eax, eax
00007D9D 7503           <1>         jnz     short loc_gfc_pass_inc_free_cluster_count
00007D9F FF4674         <1>         inc     dword [esi+LD_FreeSectors] ; Free clusters !
                        <1>
                        <1> loc_gfc_pass_inc_free_cluster_count:
                        <1>         ;mov    eax, [FAT_CurrentCluster]
00007DA2 89C8           <1>         mov     eax, ecx ; [FAT_CurrentCluster]
00007DA4 3B4678         <1>         cmp     eax, [esi+LD_Clusters]
00007DA7 77E8           <1>         ja      short retn_from_get_free_fat_clusters
00007DA9 40             <1>         inc     eax
                        <1>         ;mov    [FAT_CurrentCluster], eax
00007DAA EBDA           <1>         jmp     short loc_gfc_loop_get_next_cluster
                        <1>
                        <1> floppy_drv_init:
                        <1>         ; 09/12/2017
                        <1>         ; 06/07/2016
                        <1>         ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>         ; 24/07/2011
                        <1>         ; 04/07/2009
                        <1>         ; INPUT ->
                        <1>         ;       DL = Drive number (0,1)
                        <1>         ; OUTPUT ->
                        <1>         ;       BL = drive name
                        <1>         ;       BH = drive number
                        <1>         ;       ESI = Logical DOS drv description table
                        <1>         ;       EAX = Volume serial number
                        <1>
00007DAC BE[0E6E0000]   <1>         mov     esi, fd0_type ; 10/01/2016
00007DB1 BF00010900     <1>         mov     edi, Logical_DOSDisks
00007DB6 08D2           <1>         or      dl, dl
00007DB8 7407           <1>         jz      short loc_drv_init_fd0_fd1
00007DBA 81C700010000   <1>         add     edi, 100h
00007DC0 46             <1>         inc     esi ; fd1_type ; 10/01/2016
                        <1> loc_drv_init_fd0_fd1:
00007DC1 C6477E00       <1>         mov     byte [edi+LD_MediaChanged], 0
00007DC5 803E01         <1>         cmp     byte [esi], 1 ; type (>0 if it is existing)
                        <1>                 ; 4 = 1.44 MB, 80 track, 3 1/2"
00007DC8 7221           <1>         jb      short read_fd_boot_sector_retn
00007DCA 885702         <1>         mov     [edi+LD_PhyDrvNo], dl
                        <1> read_fd_boot_sector:
00007DCD 30F6           <1>         xor     dh, dh
00007DCF B904000000     <1>         mov     ecx, 4 ; Retry Count
                        <1> read_fd_boot_sector_again:
00007DD4 51             <1>         push    ecx
                        <1>         ;mov    cx, 1
00007DD5 B101           <1>         mov     cl, 1
00007DD7 66B80102       <1>         mov     ax, 0201h ; Read 1 sector
00007DDB BB[EE8F0100]   <1>         mov     ebx, DOSBootSectorBuff
00007DE0 E84CD4FFFF     <1>         call    int13h
00007DE5 59             <1>         pop     ecx
00007DE6 7304           <1>         jnc     short use_fd_boot_sector_params
00007DE8 E2EA           <1>         loop    read_fd_boot_sector_again
                        <1>
                        <1> read_fd_boot_sector_stc_retn:
00007DEA F9             <1>         stc
                        <1> read_fd_boot_sector_retn:
00007DEB C3             <1>         retn
                        <1>
                        <1> use_fd_boot_sector_params:
                        <1>         ;mov    esi, DOSBootSectorBuff
00007DEC 89DE           <1>         mov     esi, ebx
00007DEE 6681BEFE01000055AA <1>     cmp     word [esi+BS_Validation], 0AA55h
00007DF7 75F1           <1>         jne     short read_fd_boot_sector_stc_retn
00007DF9 66817E035346   <1>           cmp     word [esi+bs_FS_Identifier], 'SF'
00007DFF 0F85A2000000   <1>           jne     use_fd_fatfs_boot_sector_params
                        <1>         ;
00007E05 8A462D         <1>         mov     al, [esi+bs_FS_LBA_Ready]
00007E08 884705         <1>         mov     [edi+LD_FS_LBAYes], al
                        <1>         ;
                        <1>         ; 03/01/2010 CHS -> DOS FAT/BPB compatibility fix
00007E0B 8A4608         <1>         mov     al, [esi+bs_FS_MediaAttrib]
00007E0E 884706         <1>         mov     [edi+LD_FS_MediaAttrib], al
                        <1>         ;
00007E11 8A460A         <1>           mov   al, [esi+bs_FS_VersionMaj]
00007E14 884707         <1>         mov     byte [edi+LD_FS_VersionMajor], al
00007E17 668B4606       <1>         mov     ax, [esi+bs_FS_BytesPerSec]
00007E1B 66894711       <1>         mov     [edi+LD_FS_BytesPerSec], ax
00007E1F 8A462E         <1>         mov     al, [esi+bs_FS_SecPerTrack]
00007E22 28E4           <1>         sub     ah, ah ; 09/12/2017
00007E24 6689471E       <1>         mov     [edi+LD_FS_SecPerTrack], ax
00007E28 8A462F         <1>         mov     al, [esi+bs_FS_Heads]
00007E2B 66894720       <1>         mov     [edi+LD_FS_NumHeads], ax
                        <1>         ;
00007E2F 8B4628         <1>         mov     eax, [esi+bs_FS_UnDelDirD]
00007E32 894722         <1>         mov     [edi+LD_FS_UnDelDirD], eax
00007E35 8B4618         <1>         mov     eax, [esi+bs_FS_MATLocation]
00007E38 89470C         <1>         mov     [edi+LD_FS_MATLocation], eax
00007E3B 8B461C         <1>         mov     eax, [esi+bs_FS_RootDirD]
00007E3E 894708         <1>         mov     [edi+LD_FS_RootDirD], eax
00007E41 8B460C         <1>         mov     eax, [esi+bs_FS_BeginSector]
00007E44 89476C         <1>         mov     [edi+LD_FS_BeginSector], eax
00007E47 8B4610         <1>         mov     eax, [esi+bs_FS_VolumeSize]
00007E4A 894770         <1>         mov     [edi+LD_FS_VolumeSize], eax
                        <1>         ;
00007E4D 89FE           <1>         mov     esi, edi
00007E4F 8B460C         <1>         mov     eax, [esi+LD_FS_MATLocation]
                        <1>         ;add    eax, [edi+LD_FS_BeginSector]
                        <1> read_fd_MAT_sector_again:
                        <1>         ;mov    ebx, DOSBootSectorBuff
                        <1>         ;mov    ecx, 1
00007E52 B101           <1>         mov     cl, 1
00007E54 E84DAD0000     <1>         call    chs_read
00007E59 89DE           <1>         mov     esi, ebx
00007E5B 7301           <1>         jnc     short use_fdfs_mat_sector_params
```

```
1174                                    <1>       ;jmp   short read_fd_boot_sector_retn
1175 00007E5D C3                       <1>       retn
1176                                    <1> use_fdfs_mat_sector_params:
1177 00007E5E 8B460C                   <1>       mov    eax, [esi+FS_MAT_DATLocation]
1178 00007E61 894714                   <1>       mov    [edi+LD_FS_DATLocation], eax
1179 00007E64 8B4610                   <1>       mov    eax, [esi+FS_MAT_DATScount]
1180 00007E67 894718                   <1>       mov    [edi+LD_FS_DATSectors], eax
1181 00007E6A 8B4714                   <1>       mov    eax, [edi+FS_MAT_FreeSectors]
1182 00007E6D 894774                   <1>       mov    [edi+LD_FS_FreeSectors], eax
1183 00007E70 8B4618                   <1>       mov    eax, [esi+FS_MAT_FirstFreeSector]
1184 00007E73 894778                   <1>       mov    [edi+LD_FS_FirstFreeSector], eax
1185                                    <1>       ;
1186 00007E76 89FE                     <1>       mov    esi, edi
1187 00007E78 8B4608                   <1>       mov    eax, [esi+LD_FS_RootDirD]
1188                                    <1> read_fd_RDT_sector_again:
1189                                    <1>       ;mov   ebx, DOSBootSectorBuff
1190                                    <1>       ;mov   cx, 1
1191 00007E7B B101                     <1>       mov    cl, 1
1192 00007E7D E824AD0000               <1>       call   chs_read
1193 00007E82 89DE                     <1>       mov    esi, ebx
1194 00007E84 7220                     <1>       jc     short read_fd_RDT_sector_retn
1195                                    <1> use_fdfs_RDT_sector_params:
1196 00007E86 8B461C                   <1>       mov    eax, [esi+FS_RDT_VolumeSerialNo]
1197 00007E89 894728                   <1>       mov    [edi+LD_FS_VolumeSerial], eax
1198 00007E8C 57                       <1>       push   edi
1199                                    <1>       ;mov   ecx, 16
1200 00007E8D B110                     <1>       mov    cl, 16
1201 00007E8F 83C640                   <1>       add    esi, FS_RDT_VolumeName
1202 00007E92 83C72C                   <1>       add    edi, LD_FS_VolumeName
1203 00007E95 F3A5                     <1>       rep    movsd ; 64 bytes
1204 00007E97 5E                       <1>       pop    esi
1205 00007E98 C6460300                 <1>       mov    byte [esi+LD_FATType], 0
1206 00007E9C C64604A1                 <1>       mov    byte [esi+LD_FSType], 0A1h
1207 00007EA0 E9A5000000               <1>         jmp     loc_cont_use_fd_boot_sector_params
1208                                    <1>
1209                                    <1> read_fd_RDT_sector_stc_retn:
1210 00007EA5 F9                       <1>       stc
1211                                    <1> read_fd_RDT_sector_retn:
1212 00007EA6 C3                       <1>       retn
1213                                    <1>
1214                                    <1> use_fd_fatfs_boot_sector_params:
1215 00007EA7 807E2629                 <1>       cmp    byte [esi+BS_BootSig], 29h
1216 00007EAB 75F8                     <1>       jne    short read_fd_RDT_sector_stc_retn
1217 00007EAD 807E15F0                 <1>       cmp    byte [esi+BPB_Media], 0F0h
1218 00007EB1 72F3                     <1>       jb     short read_fd_RDT_sector_retn
1219 00007EB3 57                       <1>       push   edi
1220 00007EB4 83C706                   <1>       add    edi, LD_BPB
1221                                    <1>       ;mov   ecx, 16
1222 00007EB7 B110                     <1>       mov    cl, 16
1223 00007EB9 F3A5                     <1>       rep    movsd ; 64 bytes
1224 00007EBB 5E                       <1>       pop    esi
1225 00007EBC 31C0                     <1>       xor    eax, eax
1226 00007EBE 89466C                   <1>       mov    [esi+LD_StartSector], eax ; 0
1227 00007EC1 668B461C                 <1>       mov    ax, [esi+LD_BPB+BPB_FATSz16]
1228 00007EC5 8A4E16                   <1>       mov    cl, [esi+LD_BPB+BPB_NumFATs]
1229 00007EC8 F7E1                     <1>       mul    ecx
1230                                    <1>       ; edx = 0 !
1231 00007ECA 668B5614                 <1>       mov    dx, [esi+LD_BPB+BPB_RsvdSecCnt]
1232 00007ECE 66895660                 <1>       mov    [esi+LD_FATBegin], dx
1233                                    <1>       ;add   eax, edx
1234 00007ED2 6601D0                   <1>       add    ax, dx
1235 00007ED5 894664                   <1>       mov    [esi+LD_ROOTBegin], eax
1236 00007ED8 894668                   <1>       mov    [esi+LD_DATABegin], eax
1237 00007EDB 668B5617                 <1>       mov    dx, [esi+LD_BPB+BPB_RootEntCnt]
1238                                    <1>       ;;shl  edx, 5 ; * 32 (Size of a directory entry)
1239                                    <1>       ;shl   dx, 5
1240                                    <1>       ;;add  edx, 511
1241                                    <1>       ;add   dx, 511
1242                                    <1>       ;;shr  edx, 9 ; edx = ((edx*32)+511) / 512
1243                                    <1>       ;shr   dx, 9
1244 00007EDF 6683C20F                 <1>       add    dx, 15 ; 06/07/2016 (+(512/32)-1)
1245 00007EE3 66C1EA04                 <1>       shr    dx, 4 ; / 16 (==16 entries per sector)
1246 00007EE7 015668                   <1>       add    [esi+LD_DATABegin], edx ; + rd sectors
1247                                    <1>       ;movzx eax, word [esi+LD_BPB+BPB_TotalSec16]
1248 00007EEA 668B4619                 <1>       mov    ax, [esi+LD_BPB+BPB_TotalSec16]
1249 00007EEE 894670                   <1>       mov    [esi+LD_TotalSectors], eax
1250 00007EF1 2B4668                   <1>       sub    eax, [esi+LD_DATABegin]
1251                                    <1>       ;movzx ecx, byte [esi+LD_BPB+BPB_SecPerClust]
1252 00007EF4 8A4E13                   <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
1253 00007EF7 80F901                   <1>       cmp    cl, 1
1254 00007EFA 7605                     <1>       jna    short save_fd_fatfs_cluster_count
1255                                    <1>       ;sub   edx, edx
1256 00007EFC 6629D2                   <1>       sub    dx, dx ; 0
1257                                    <1>       ;sub   dl, dl ; 06/07/2016
1258 00007EFF F7F1                     <1>       div    ecx
1259                                    <1> save_fd_fatfs_cluster_count:
1260 00007F01 894678                   <1>       mov    [esi+LD_Clusters], eax
1261                                    <1>
1262                                    <1>       ; Maximum Valid Cluster Number = EAX +1
1263                                    <1>       ; with 2 reserved clusters= EAX +2
1264                                    <1>
1265                                    <1> reset_FAT_buffer_decriptors:
1266 00007F04 29C0                     <1>       sub    eax, eax ; 0
1267 00007F06 A2[F2910100]             <1>       mov    [FAT_BuffValidData], al ; 0
1268 00007F0B A2[F3910100]             <1>       mov    [FAT_BuffDrvName], al ; 0
1269 00007F10 A3[F6910100]             <1>       mov    [FAT_BuffSector], eax ; 0
1270                                    <1>
1271                                    <1> read_fd_FAT_sectors:
1272 00007F15 BB001C0900               <1>       mov    ebx, FAT_Buffer
1273 00007F1A 668B4614                 <1>       mov    ax, [esi+LD_BPB+BPB_RsvdSecCnt]
1274                                    <1>       ;mov   ecx, 3
1275 00007F1E B103                     <1>       mov    cl, 3 ; 3 sectors
1276 00007F20 E881AC0000               <1>       call   chs_read
1277 00007F25 7240                     <1>       jc     short read_fd_FAT_sectors_retn
1278                                    <1> use_fd_FAT_sectors:
```

```
1279 00007F27 8A4602              <1>       mov    al, [esi+LD_PhyDrvNo]
1280 00007F2A 0441                <1>       add    al, 'A'
1281 00007F2C A2[F3910100]        <1>       mov    [FAT_BuffDrvName], al
1282 00007F31 C605[F2910100]01    <1>       mov    byte [FAT_BuffValidData], 1
1283 00007F38 E82B000000          <1>       call   fd_init_calculate_free_clusters
1284 00007F3D 7228                <1>       jc     short read_fd_FAT_sectors_retn
1285                               <1>
1286                               <1> loc_use_fd_boot_sector_params_FAT:
1287 00007F3F C6460301            <1>       mov    byte [esi+LD_FATType], 1 ; FAT 12
1288 00007F43 C6460401            <1>       mov    byte [esi+LD_FSType], 1
1289 00007F47 8B462D              <1>         mov     eax, [esi+LD_BPB+VolumeID]
1290                               <1> loc_cont_use_fd_boot_sector_params:
1291 00007F4A 8A7E02              <1>       mov    bh, [esi+LD_PhyDrvNo]
1292 00007F4D 887E7D              <1>       mov    [esi+LD_DParamEntry], bh
1293 00007F50 88FB                <1>       mov    bl, bh
1294 00007F52 80C341              <1>       add    bl, 'A'
1295 00007F55 881E                <1>       mov    byte [esi+LD_Name], bl
1296 00007F57 C6460101            <1>       mov    byte [esi+LD_DiskType], 1
1297 00007F5B C6460500            <1>       mov    byte [esi+LD_LBAYes], 0
1298 00007F5F C6467C00            <1>       mov    byte [esi+LD_PartitionEntry], 0
1299 00007F63 C6467E06            <1>       mov    byte [esi+LD_MediaChanged], 6 ; Volume Name Reset
1300                               <1>
1301                               <1> read_fd_FAT_sectors_retn:
1302 00007F67 C3                  <1>       retn
1303                               <1>
1304                               <1> fd_init_calculate_free_clusters:
1305                               <1>       ; 09/12/2017
1306                               <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1307                               <1>       ; 04/07/2009
1308                               <1>       ; INPUT ->
1309                               <1>       ;     ESI = Logical DOS drive description table address
1310                               <1>       ; OUTPUT ->
1311                               <1>       ;     [ESI+LD_FreeSectors] will be set
1312                               <1>
1313 00007F68 29C0                <1>       sub    eax, eax
1314 00007F6A 894674              <1>       mov    [esi+LD_FreeSectors], eax ; 0
1315 00007F6D B002                <1>       mov    al, 2 ; eax = 2
1316                               <1>
1317                               <1> fd_init_loop_get_next_cluster:
1318 00007F6F E830000000          <1>       call   fd_init_get_next_cluster
1319 00007F74 722D                <1>       jc     short fd_init_calculate_free_clusters_retn
1320                               <1>
1321                               <1> fd_init_free_fat_clusters:
1322                               <1>       ;cmp   eax, 0
1323                               <1>       ;ja    short fd_init_pass_inc_free_cluster_count
1324                               <1>       ;and   eax, eax
1325                               <1>       ;jnz   short fd_init_pass_inc_free_cluster_count
1326 00007F76 6621C0              <1>       and    ax, ax
1327 00007F79 7504                <1>       jnz    short fd_init_pass_inc_free_cluster_count
1328                               <1>       ;inc   dword [esi+LD_FreeSectors]
1329 00007F7B 66FF4674            <1>         inc  word [esi+LD_FreeSectors]
1330                               <1>
1331                               <1> fd_init_pass_inc_free_cluster_count:
1332                               <1>       ;mov   eax, [FAT_CurrentCluster]
1333 00007F7F 66A1[EE910100]      <1>       mov    ax, [FAT_CurrentCluster]
1334                               <1>       ;cmp   eax, [esi+LD_Clusters]
1335 00007F85 663B4678            <1>       cmp    ax, [esi+LD_Clusters]
1336 00007F89 7704                <1>       ja     short short retn_from_fd_init_calculate_free_clusters
1337                               <1>       ;inc   eax
1338 00007F8B 6640                <1>       inc    ax
1339 00007F8D EBE0                <1>       jmp    short fd_init_loop_get_next_cluster
1340                               <1>
1341                               <1> retn_from_fd_init_calculate_free_clusters:
1342 00007F8F 8A4613              <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust]
1343 00007F92 3C01                <1>       cmp    al, 1
1344 00007F94 760D                <1>       jna    short fd_init_calculate_free_clusters_retn
1345                               <1>       ;movzx eax, al
1346 00007F96 30E4                <1>       xor    ah, ah ; 09/12/2017
1347                               <1>       ;mov   ecx, [esi+LD_FreeSectors]
1348 00007F98 668B4E74            <1>       mov    cx, [esi+LD_FreeSectors] ; Count of free clusters
1349                               <1>       ;mul   ecx
1350 00007F9C 66F7E1              <1>       mul    cx
1351                               <1>       ;mov   [esi+LD_FreeSectors], eax
1352 00007F9F 66894674            <1>       mov    [esi+LD_FreeSectors], ax
1353                               <1> fd_init_calculate_free_clusters_retn:
1354 00007FA3 C3                  <1>       retn
1355                               <1>
1356                               <1> fd_init_get_next_cluster:
1357                               <1>       ; 04/02/2016
1358                               <1>       ; 02/02/2016
1359                               <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1360                               <1>       ; 04/07/2009
1361                               <1>       ; INPUT ->
1362                               <1>       ;    EAX = Current cluster
1363                               <1>       ;    ESI = Logical DOS drive description table address
1364                               <1>       ;    EDX = 0
1365                               <1>       ; OUTPUT ->
1366                               <1>       ;    EAX = Next cluster
1367                               <1>
1368 00007FA4 A3[EE910100]        <1>       mov    [FAT_CurrentCluster], eax
1369                               <1> fd_init_get_next_cluster_readnext:
1370 00007FA9 29D2                <1>       sub    edx, edx ; 0
1371 00007FAB BB00040000          <1>       mov    ebx, 1024 ; 400h
1372 00007FB0 F7F3                <1>       div    ebx
1373                               <1>       ; EAX = Count of 3 FAT sectors
1374                               <1>       ; EDX = Buffer entry index
1375 00007FB2 89C1                <1>       mov    ecx, eax
1376                               <1>       ;mov   eax, 3
1377 00007FB4 B003                <1>       mov    al, 3
1378 00007FB6 F7E2                <1>       mul    edx ; Multiply by 3
1379 00007FB8 66D1E8              <1>       shr    ax, 1 ; Divide by 2
1380 00007FBB 89C3                <1>       mov    ebx, eax ; Buffer byte offset
1381 00007FBD 81C3001C0900        <1>       add    ebx, FAT_Buffer
1382 00007FC3 89C8                <1>       mov    eax, ecx
1383                               <1>       ;mov   edx, 3
```

```
1384 00007FC5 66BA0300            <1>       mov    dx, 3
1385 00007FC9 F7E2                <1>       mul    edx
1386                              <1>       ; EAX = FAT Beginning Sector
1387                              <1>       ; EDX = 0
1388 00007FCB 8A0E                <1>       mov    cl, [esi+LD_Name]
1389                              <1>       ;cmp   byte [FAT_BuffValidData], 0
1390                              <1>       ;jna   short fd_init_load_FAT_sectors0
1391 00007FCD 3A0D[F3910100]      <1>       cmp    cl, [FAT_BuffDrvName]
1392 00007FD3 751E                <1>       jne    short fd_init_load_FAT_sectors0
1393 00007FD5 3B05[F6910100]      <1>       cmp    eax, [FAT_BuffSector]
1394 00007FDB 751C                <1>       jne    short fd_init_load_FAT_sectors1
1395                              <1>       ;mov   eax, [FAT_CurrentCluster]
1396 00007FDD A0[EE910100]        <1>       mov    al, [FAT_CurrentCluster]
1397                              <1>       ;shr   eax, 1
1398 00007FE2 D0E8                <1>       shr    al, 1
1399 00007FE4 668B03              <1>       mov    ax, [ebx]
1400 00007FE7 7306                <1>       jnc    short fd_init_gnc_even
1401 00007FE9 66C1E804            <1>       shr    ax, 4
1402                              <1> fd_init_gnc_clc_retn:
1403 00007FED F8                  <1>       clc
1404 00007FEE C3                  <1>       retn
1405                              <1>
1406                              <1> fd_init_gnc_even:
1407 00007FEF 80E40F              <1>       and    ah, 0Fh
1408 00007FF2 C3                  <1>       retn
1409                              <1>
1410                              <1> fd_init_load_FAT_sectors0:
1411 00007FF3 880D[F3910100]      <1>       mov    [FAT_BuffDrvName], cl
1412                              <1> fd_init_load_FAT_sectors1:
1413 00007FF9 C605[F2910100]00    <1>       mov    byte [FAT_BuffValidData], 0
1414 00008000 A3[F6910100]        <1>       mov    [FAT_BuffSector], eax
1415 00008005 034660              <1>       add    eax, [esi+LD_FATBegin]
1416 00008008 BB001C0900          <1>       mov    ebx, FAT_Buffer
1417                              <1>       ;movzx ecx, word [esi+LD_BPB+BPB_FATSz16]
1418 0000800D 668B4E1C            <1>       mov    cx, [esi+LD_BPB+BPB_FATSz16]
1419 00008011 662B0D[F6910100]    <1>       sub    cx, [FAT_BuffSector]
1420                              <1>         ;cmp ecx, 3
1421 00008018 6683F903            <1>       cmp    cx, 3
1422 0000801C 7605                <1>       jna    short fdinit_pass_fix_sector_count_3
1423                              <1>       ;mov   ecx, 3
1424 0000801E B903000000          <1>       mov    ecx, 3
1425                              <1> fdinit_pass_fix_sector_count_3:
1426 00008023 E87EAB0000          <1>       call   chs_read
1427 00008028 730D                <1>       jnc    short fd_init_FAT_sectors_no_load_error
1428 0000802A C605[F2910100]00    <1>       mov    byte [FAT_BuffValidData], 0
1429                              <1>              ; Drv not ready or read Error !
1430 00008031 B80F000000          <1>       mov    eax, ERR_DRV_NOT_RDY ; 15
1431                              <1>       ;xor   edx, edx
1432 00008036 C3                  <1>       retn
1433                              <1>
1434                              <1> fd_init_FAT_sectors_no_load_error:
1435 00008037 C605[F2910100]01    <1>       mov    byte [FAT_BuffValidData], 1
1436 0000803E A1[EE910100]        <1>       mov    eax, [FAT_CurrentCluster]
1437 00008043 E961FFFFFF          <1>         jmp     fd_init_get_next_cluster_readnext
1438                              <1>
1439                              <1> get_FAT_volume_name:
1440                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1441                              <1>       ; 12/09/2009
1442                              <1>       ; INPUT ->
1443                              <1>       ;      BH = Logical DOS drive number (0,1,2,3,4 ...)
1444                              <1>       ;       BL = 0
1445                              <1>       ; OUTPUT ->
1446                              <1>       ;      CF = 0 -> ESI = Volume name address
1447                              <1>       ;      CF = 1 -> Root volume name not found
1448                              <1>
1449                              <1>       ;mov   ah, 0FFh
1450                              <1>       ;mov   al, [Last_Dos_DiskNo]
1451                              <1>       ;cmp   al, bh
1452                              <1>       ;jb     short loc_gfvn_dir_load_err
1453                              <1>
1454 00008048 89DE                <1>       mov    esi, ebx
1455 0000804A 81E600FF0000        <1>       and    esi, 0FF00h ; esi = bh
1456 00008050 81C600010900        <1>       add    esi, Logical_DOSDisks
1457 00008056 8A06                <1>       mov     al, [esi+LD_Name]
1458 00008058 8A6603              <1>       mov     ah, [esi+LD_FATType]
1459 0000805B 80FC01              <1>       cmp     ah, 1
1460 0000805E 7210                <1>       jb     short loc_gfvn_dir_load_err
1461 00008060 3C41                <1>       cmp    al, 'A'
1462 00008062 720C                <1>       jb      short loc_gfvn_dir_load_err
1463 00008064 80FC02              <1>       cmp    ah, 2
1464 00008067 7708                <1>       ja      short get_FAT32_root_cluster
1465                              <1>
1466 00008069 E8634E0000          <1>       call    load_FAT_root_directory
1467 0000806E 730B                <1>       jnc     short loc_get_volume_name
1468                              <1>
1469                              <1> loc_gfvn_dir_load_err:
1470 00008070 C3                  <1>       retn
1471                              <1>
1472                              <1> get_FAT32_root_cluster:
1473 00008071 8B4632              <1>       mov    eax, [esi+LD_BPB+BPB_RootClus]
1474 00008074 E8E34E0000          <1>       call    load_FAT_sub_directory
1475 00008079 7224                <1>       jc     short loc_get_volume_name_retn
1476                              <1>
1477                              <1> loc_get_volume_name:
1478 0000807B BE00000800          <1>         mov     esi, Directory_Buffer
1479 00008080 6631C9              <1>       xor    cx, cx ; 0
1480                              <1> check_root_volume_name:
1481 00008083 8A06                <1>       mov    al, [esi]
1482 00008085 08C0                <1>       or      al, al
1483 00008087 7416                <1>       jz      short loc_get_volume_name_retn
1484 00008089 807E0B08            <1>       cmp     byte [esi+0Bh], 08h
1485 0000808D 7410                <1>       je      short loc_get_volume_name_retn
1486 0000808F 663B0D[07920100]    <1>       cmp     cx, [DirBuff_LastEntry]
1487 00008096 7308                <1>       jnb     short pass_check_root_volume_name
1488 00008098 6641                <1>       inc     cx
```

```
1489 0000809A 83C620              <1>       add     esi, 32
1490 0000809D EBE4                <1>       jmp     short check_root_volume_name
1491                              <1>
1492                              <1> loc_get_volume_name_retn:
1493 0000809F C3                  <1>       retn
1494                              <1>
1495                              <1> pass_check_root_volume_name:
1496 000080A0 803D[03920100]03    <1>       cmp    byte [DirBuff_FATType], 3
1497 000080A7 7230                <1>       jb     short loc_get_volume_name_retn_xor
1498                              <1>
1499 000080A9 BB001C0900          <1>       mov    ebx, FAT_Buffer
1500 000080AE BE00010900          <1>       mov    esi, Logical_DOSDisks
1501 000080B3 31C0                <1>       xor    eax, eax
1502 000080B5 8A25[02920100]      <1>       mov    ah, [DirBuff_DRV]
1503 000080BB 80EC41              <1>       sub    ah, 'A'
1504 000080BE 01C6                <1>       add    esi, eax
1505 000080C0 A1[09920100]        <1>       mov    eax, [DirBuff_Cluster]
1506 000080C5 E8AC4C0000          <1>       call   get_next_cluster
1507 000080CA 7305                <1>       jnc    short loc_gfvn_load_FAT32_dir_cluster
1508                              <1>
1509 000080CC 83F801              <1>       cmp     eax, 1
1510 000080CF F5                  <1>       cmc
1511 000080D0 C3                  <1>       retn
1512                              <1>
1513                              <1> loc_gfvn_load_FAT32_dir_cluster:
1514 000080D1 E8864E0000          <1>       call   load_FAT_sub_directory
1515 000080D6 73A3                <1>       jnc    short loc_get_volume_name
1516 000080D8 C3                  <1>       retn
1517                              <1>
1518                              <1> loc_get_volume_name_retn_xor:
1519 000080D9 31C0                <1>       xor    eax, eax
1520 000080DB C3                  <1>       retn
1521                              <1>
1522                              <1> get_media_change_status:
1523                              <1>       ; 10/01/2016 (TRDOS 386 = TRDOS v2.0)
1524                              <1>       ; 09/09/2009
1525                              <1>       ; INPUT:
1526                              <1>       ;     DL = Drive number (physical)
1527                              <1>       ; OUTPUT: clc & AH = 6 media changed
1528                              <1>       ;     clc & AH = 0 media not changed
1529                              <1>       ;     stc -> Drive not ready or an error
1530                              <1>
1531 000080DC B416                <1>       mov    ah, 16h
1532 000080DE E84ED1FFFF          <1>       call   int13h
1533 000080E3 80FC06              <1>       cmp    ah, 06h
1534 000080E6 7405                <1>       je     short loc_gmc_status_retn
1535 000080E8 08E4                <1>       or     ah, ah
1536 000080EA 7401                <1>       jz     short loc_gmc_status_retn
1537                              <1> loc_gmc_status_stc_retn:
1538 000080EC F9                  <1>       stc
1539                              <1> loc_gmc_status_retn:
1540 000080ED C3                  <1>       retn
3089                                  %include 'trdosk3.s' ; 06/01/2016
   1                              <1> ; ****************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - MAIN PROGRAM : trdosk3.s
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 31/12/2017
   5                              <1> ; ----------------------------------------------------------------------------
   6                              <1> ; Beginning: 06/01/2016
   7                              <1> ; ----------------------------------------------------------------------------
   8                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                              <1> ; ----------------------------------------------------------------------------
  10                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
  11                              <1> ; MAINPROG.ASM (09/11/2011)
  12                              <1> ; ****************************************************************************
  13                              <1> ; MAINPROG.ASM [ TRDOS KERNEL - COMMAND EXECUTER SECTION - MAIN PROGRAM ]
  14                              <1> ; (c) 2004-2011  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/11/2011
  15                              <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ] Last Update: 09/11/2011
  16                              <1> ; DIR.ASM [ DIRECTORY FUNCTIONS ] Last Update: 09/10/2011
  17                              <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
  18                              <1>
  19                              <1> change_current_drive:
  20                              <1>       ; 16/10/2016
  21                              <1>       ; 02/02/2016
  22                              <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
  23                              <1>       ; 18/08/2011
  24                              <1>       ; 09/09/2009
  25                              <1>       ; INPUT:
  26                              <1>       ;   DL = Logical DOS Drive Number
  27                              <1>       ; OUTPUT:
  28                              <1>       ;  cf=1 -> Not successful
  29                              <1>       ;   EAX = Error code
  30                              <1>       ;  cf=0 ->
  31                              <1>       ;   EAX = 0 (successful)
  32                              <1>
  33 000080EE 31DB                <1>       xor    ebx, ebx
  34 000080F0 88D7                <1>       mov    bh, dl
  35                              <1>
  36                              <1>       ;cmp   dl, 1
  37                              <1>       ;jna   short loc_ccdrv_initial_media_change_check
  38                              <1>       ;cmp   bh, [Last_Dos_DiskNo]
  39                              <1>       ;ja    short loc_ccdrv_drive_not_ready_err
  40                              <1>
  41                              <1> loc_ccdrv_initial_media_change_check:
  42 000080F2 BE00010900          <1>       mov    esi, Logical_DOSDisks
  43 000080F7 01DE                <1>       add    esi, ebx
  44                              <1> loc_ccdrv_dos_drive_name_check:
  45 000080F9 80FA02              <1>       cmp    dl, 2
  46 000080FC 720F                <1>       jb     short loc_ccdrv_dos_drive_name_check_ok
  47                              <1>
  48 000080FE 8A06                <1>       mov    al, [esi+LD_Name]
  49 00008100 2C41                <1>       sub    al, 'A'
  50 00008102 38D0                <1>       cmp    al, dl
  51 00008104 7407                <1>       je     short loc_ccdrv_dos_drive_name_check_ok
  52                              <1>
```

```
                        <1> loc_ccdrv_drive_not_ready_err:
                        <1>       ; 16/10/2016 (15h -> 15)
00008106 B80F000000     <1>       mov    eax, 15 ; Drive not ready
                        <1> loc_change_current_drive_stc_retn:
0000810B F9             <1>       stc
0000810C C3             <1>       retn
                        <1>
                        <1> loc_ccdrv_dos_drive_name_check_ok:
0000810D 8A667E         <1>       mov    ah, [esi+LD_MediaChanged]
00008110 80FC06         <1>       cmp    ah, 6  ; VOLUME NAME CHECK/MOVE SIGN
00008113 7455           <1>       je     short loc_ccdrv_get_FAT_volume_name_0
                        <1>
00008115 80FA01         <1>       cmp    dl, 1
00008118 777D           <1>       ja     short loc_gmcs_init_drv_hd
                        <1>
                        <1> loc_gmcs_init_drv_fd:
0000811A 08E4           <1>       or     ah, ah
                        <1>       ; AH = 1 is initialization sign (invalid_fd_parameter)
0000811C 7517           <1>       jnz    short loc_ccdrv_call_fd_init
                        <1>
0000811E E8B9FFFFFF     <1>       call   get_media_change_status
00008123 72E1           <1>       jc     short loc_ccdrv_drive_not_ready_err
                        <1>
00008125 20E4           <1>       and    ah, ah
00008127 7476           <1>       jz     short loc_change_current_drv3
                        <1>
00008129 80F406         <1>       xor    ah, 6
0000812C 75D8           <1>       jnz    short loc_ccdrv_drive_not_ready_err
                        <1>
                        <1> loc_ccdrv_call_fd_init_check_vol_id:
0000812E E8440A0000     <1>       call   get_volume_serial_number
00008133 730D           <1>       jnc    short loc_ccdrv_check_vol_serial
                        <1>
                        <1> loc_ccdrv_call_fd_init:
00008135 E872FCFFFF     <1>       call   floppy_drv_init
0000813A 731A           <1>       jnc    short loc_reset_drv_fd_current_dir
                        <1>
                        <1> loc_ccdrv_fdinit_fail_retn:
                        <1>       ; 16/10/2016
0000813C B80F000000     <1>       mov    eax, 15 ; Drive not ready
00008141 C3             <1>       retn
                        <1>
                        <1> loc_ccdrv_check_vol_serial:
00008142 A3[D48A0100]   <1>       mov    [Current_VolSerial], eax
                        <1>       ;mov   dl, bh
00008147 E860FCFFFF     <1>       call   floppy_drv_init
0000814C 72EE           <1>       jc     short loc_ccdrv_fdinit_fail_retn
                        <1>
0000814E 3B05[D48A0100] <1>       cmp    eax, [Current_VolSerial]
00008154 7445           <1>       je     short loc_change_current_drv2
                        <1>
                        <1> loc_reset_drv_fd_current_dir:
00008156 31C0           <1>       xor    eax, eax
00008158 88467F         <1>         mov  [esi+LD_CDirLevel], al
0000815B 89F7           <1>       mov    edi, esi
0000815D 81C780000000   <1>       add    edi, LD_CurrentDirectory
00008163 B920000000     <1>       mov    ecx, 32
00008168 F3AB           <1>       rep    stosd
                        <1>
                        <1> loc_ccdrv_get_FAT_volume_name_0:
0000816A 8A4603         <1>       mov    al, [esi+LD_FATType]
0000816D 08C0           <1>       or     al, al
0000816F 742A           <1>       jz     short loc_change_current_drv2
                        <1>
00008171 56             <1>       push   esi
00008172 3C02           <1>       cmp    al, 2
00008174 7705           <1>       ja     short loc_ccdrv_get_FAT32_vol_name
                        <1>
                        <1> loc_ccdrv_get_FAT2_16_vol_name:
00008176 83C631         <1>       add    esi, LD_BPB + VolumeLabel
00008179 EB03           <1>       jmp    short loc_ccdrv_get_FAT_volume_name_1
                        <1>
                        <1> loc_ccdrv_get_FAT32_vol_name:
0000817B 83C64D         <1>       add    esi, LD_BPB + FAT32_VolLab
                        <1> loc_ccdrv_get_FAT_volume_name_1:
0000817E 53             <1>       push   ebx
0000817F 56             <1>       push   esi
00008180 E8C3FEFFFF     <1>       call   get_FAT_volume_name
00008185 5F             <1>       pop    edi
00008186 5B             <1>       pop    ebx
                        <1>       ; BL = 0
00008187 720B           <1>       jc     short loc_change_current_drv1
00008189 20C0           <1>       and    al, al
0000818B 7407           <1>       jz     short loc_change_current_drv1
                        <1>
                        <1> loc_ccdrv_move_FAT_volume_name:
0000818D B90B000000     <1>       mov    ecx, 11
00008192 F3A4           <1>       rep    movsb
                        <1>
                        <1> loc_change_current_drv1:
00008194 5E             <1>       pop    esi
00008195 EB04           <1>       jmp    short loc_change_current_drv2
                        <1>
                        <1> loc_gmcs_init_drv_hd:
00008197 08E4           <1>       or     ah, ah
00008199 7404           <1>       jz     short loc_change_current_drv3
                        <1>       ; BL = 0, BH = Logical DOS drive number
                        <1> loc_change_current_drv2:
0000819B C6467E00       <1>       mov    byte [esi+LD_MediaChanged], 0
                        <1> loc_change_current_drv3:
0000819F 883D[DE8A0100] <1>       mov    [Current_Drv], bh
                        <1>
                        <1>       ;call  restore_current_directory
                        <1>       ;retn
                        <1>
```

```
                            <1> restore_current_directory:
                            <1>       ; 11/02/2016
                            <1>       ; 15/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 25/01/2010
                            <1>       ; 12/10/2009
                            <1>       ;
                            <1>       ; INPUT:
                            <1>       ;   ESI = Logical DOS Drive Description Table
                            <1>       ;
                            <1>       ; OUTPUT:
                            <1>       ;   ESI = Logical DOS Drive Description Table
                            <1>       ;   EDI = offset Current_Dir_Drv
                            <1>
000081A5 8A4603             <1>       mov   al, [esi+LD_FATType]
000081A8 A2[DD8A0100]       <1>       mov   [Current_FATType], al
                            <1>
000081AD 8A26               <1>       mov   ah, [esi+LD_Name]
000081AF 8825[DF8A0100]     <1>       mov   [Current_Dir_Drv], ah
                            <1>
000081B5 20C0               <1>       and   al, al
000081B7 741D               <1>       jz    short loc_restore_FS_current_directory
                            <1>
                            <1> loc_restore_FAT_current_directory:
000081B9 8A667F             <1>       mov   ah, [esi+LD_CDirLevel]
000081BC 8825[DC8A0100]     <1>       mov   [Current_Dir_Level], ah
000081C2 08E4               <1>       or    ah, ah
000081C4 7416               <1>         jz  short loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster
                            <1>
000081C6 0FB6D4             <1>       movzx edx, ah
000081C9 C0E204             <1>       shl   dl, 4 ; * 16
000081CC 01F2               <1>         add edx, esi
000081CE 8B828C000000       <1>       mov   eax, [edx+LD_CurrentDirectory+12]
000081D4 EB2C               <1>       jmp   short loc_ccdrv_reset_cdir_FAT_fcluster
                            <1>
                            <1> loc_restore_FS_current_directory:
000081D6 E8BC4D0000         <1>       call  load_current_FS_directory
000081DB C3                 <1>       retn
                            <1>
                            <1> loc_ccdrv_reset_cdir_FAT_12_16_32_fcluster:
000081DC 3C03               <1>       cmp   al, 3
000081DE 7205               <1>       jb    short loc_ccdrv_reset_cdir_FAT_12_16_fcluster
                            <1> loc_ccdrv_reset_cdir_FAT32_fcluster:
000081E0 8B4632             <1>       mov   eax, [esi+LD_BPB+FAT32_RootFClust]
000081E3 EB04               <1>       jmp   short loc_ccdrv_check_rootdir_sign
                            <1> loc_ccdrv_reset_cdir_FAT_12_16_fcluster:
000081E5 30C0               <1>       xor   al, al  ; xor eax, eax
000081E7 31D2               <1>       xor   edx, edx
                            <1> loc_ccdrv_check_rootdir_sign:
000081E9 80BE8000000000     <1>       cmp   byte [esi+LD_CurrentDirectory], 0
000081F0 7510               <1>       jne   short loc_ccdrv_reset_cdir_FAT_fcluster
                            <1> loc_ccdrv_set_rootdir_FAT_fcluster:
000081F2 89868C000000       <1>         mov     [esi+LD_CurrentDirectory+12], eax
000081F8 C78680000000524F4F- <1>       mov   dword [esi+LD_CurrentDirectory], 'ROOT'
00008201 54                 <1>
                            <1>
                            <1> loc_ccdrv_reset_cdir_FAT_fcluster:
00008202 A3[D88A0100]       <1>       mov   [Current_Dir_FCluster], eax
                            <1>
00008207 BF[3B920100]       <1>       mov   edi, PATH_Array
0000820C 89F2               <1>       mov   edx, esi
0000820E 81C680000000       <1>       add   esi, LD_CurrentDirectory
00008214 B920000000         <1>       mov   ecx, 32
00008219 F3A5               <1>       rep   movsd
                            <1>
0000821B E84C2D0000         <1>       call  change_prompt_dir_string
                            <1>
00008220 89D6               <1>       mov   esi, edx
                            <1>
00008222 29C0               <1>         sub  eax, eax
                            <1>        ;sub  edx, edx
00008224 BF[DF8A0100]       <1>       mov   edi, Current_Dir_Drv
                            <1>
00008229 A2[A6400100]       <1>       mov   [Restore_CDIR], al ; 0
0000822E 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
0000822F C705[98960100]-    <1>       mov   dword [mainprog_return_addr], return_from_cmd_interpreter
00008235 [E3820000]         <1>
                            <1>
                            <1> loc_TRDOS_prompt:
00008239 BF[DE8B0100]       <1>       mov   edi, TextBuffer
0000823E C6075B             <1>       mov   byte [edi], "["
00008241 47                 <1>       inc   edi
00008242 BE[F9400100]       <1>       mov   esi, TRDOSPromptLabel
                            <1> get_next_prompt_label_char:
00008247 803E20             <1>       cmp   byte [esi], 20h
0000824A 7203               <1>       jb    short pass_prompt_label
0000824C A4                 <1>       movsb
0000824D EBF8               <1>       jmp   short get_next_prompt_label_char
                            <1> pass_prompt_label:
0000824F C6075D             <1>       mov   byte [edi], "]"
00008252 47                 <1>       inc   edi
00008253 C60720             <1>       mov   byte [edi], 20h
00008256 47                 <1>       inc   edi
00008257 BE[DF8A0100]       <1>       mov   esi, Current_Dir_Drv
0000825C 66A5               <1>       movsw
```

```
                                <1>       movsb
                                <1> loc_prompt_current_directory:
0000825F 803E20                 <1>       cmp   byte [esi], 20h
00008262 7203                   <1>       jb    short pass_prompt_current_directory
00008264 A4                     <1>       movsb
00008265 EBF8                   <1>       jmp   short loc_prompt_current_directory
                                <1> pass_prompt_current_directory:
00008267 C6073E                 <1>       mov   byte [edi], '>'
0000826A 47                     <1>       inc   edi
0000826B C60700                 <1>       mov   byte [edi], 0
0000826E BE[DE8B0100]           <1>       mov   esi, TextBuffer
00008273 E8EDF2FFFF             <1>       call  print_msg
                                <1>
                                <1>       ;sub   bh, bh ; video page = 0
                                <1>       ;call  get_cpos ; get cursor position
00008278 668B15[368A0100]       <1>       mov   dx, [CURSOR_POSN] ; video page 0
0000827F 8815[3E8B0100]         <1>       mov   [CursorColumn], dl
                                <1>
                                <1>       ; 30/01/2016 (to show cursor on the row, again)
                                <1>       ; (Initial color attributes of video page 0 is 0)
                                <1>       ; (see: 'StartPMP' in trdos386.s)
                                <1>       ;
                                <1>       ;mov   edi, 0B8000h ; start of video page 0
                                <1>       ;movzx ecx, dl ; column
                                <1>       ;mov   al, 80
                                <1>       ;mul   dh
                                <1>       ;add   ax, cx
                                <1>       ;shl   ax, 1 ; character + attribute
                                <1>       ;add   di, ax ; (2*80*row) + (2*column)
                                <1>       ;neg   cl
                                <1>       ;add   cl, 80
                                <1>       ;mov   ax, 700h ;  ah = 7 (color attribute)
                                <1>       ;rep   stosw
                                <1>
                                <1> loc_rw_char:
00008285 E899000000             <1>       call  rw_char
                                <1> loc_move_command:
0000828A BE[8E8B0100]           <1>       mov   esi, CommandBuffer
0000828F 89F7                   <1>       mov   edi, esi
00008291 31C9                   <1>       xor   ecx, ecx
                                <1> first_command_char:
00008293 AC                     <1>       lodsb
00008294 3C20                   <1>       cmp   al, 20h
00008296 772E                   <1>       ja    short pass_space_control
00008298 7241                   <1>       jb    short loc_move_cmd_arguments_ok
0000829A 81FE[DD8B0100]         <1>       cmp   esi, CommandBuffer + 79
000082A0 72F1                   <1>       jb    short first_command_char
000082A2 EB37                   <1>       jmp   short loc_move_cmd_arguments_ok
                                <1>
                                <1> next_command_char:
000082A4 AC                     <1>       lodsb
000082A5 3C20                   <1>       cmp   al, 20h
000082A7 771D                   <1>       ja    short pass_space_control
000082A9 7230                   <1>       jb    short loc_move_cmd_arguments_ok
                                <1>
                                <1> loc_1st_cmd_arg: ; 30/01/2016
000082AB AC                     <1>       lodsb
000082AC 3C20                   <1>       cmp   al, 20h
000082AE 74FB                   <1>       je    short loc_1st_cmd_arg
000082B0 7229                   <1>       jb    short loc_move_cmd_arguments_ok
                                <1>
000082B2 C60700                 <1>         mov     byte [edi], 0
000082B5 47                     <1>       inc   edi
                                <1>
                                <1> loc_move_cmd_arguments:
000082B6 AA                     <1>       stosb
000082B7 81FE[DD8B0100]         <1>       cmp   esi, CommandBuffer + 79
000082BD 731C                   <1>       jnb   short loc_move_cmd_arguments_ok
000082BF AC                     <1>         lodsb
000082C0 3C20                   <1>       cmp   al, 20h
000082C2 73F2                   <1>       jnb   short loc_move_cmd_arguments
000082C4 EB15                   <1>       jmp   short loc_move_cmd_arguments_ok
                                <1>
                                <1> pass_space_control:
000082C6 3C61                   <1>       cmp   al, 61h
000082C8 7206                   <1>       jb    short pass_capitalize
000082CA 3C7A                   <1>       cmp   al, 7Ah
000082CC 7702                   <1>       ja    short pass_capitalize
000082CE 24DF                   <1>       and   al, 0DFh
                                <1> pass_capitalize:
000082D0 AA                     <1>       stosb
000082D1 FEC1                   <1>       inc     cl
000082D3 81FE[DD8B0100]         <1>         cmp     esi, CommandBuffer + 79
000082D9 72C9                   <1>       jb      short next_command_char
                                <1>
                                <1> loc_move_cmd_arguments_ok:
000082DB C60700                 <1>         mov     byte [edi], 0
                                <1>
                                <1> call_command_interpreter:
000082DE E8CF080000             <1>       call    command_interpreter
                                <1>
                                <1> return_from_cmd_interpreter:
000082E3 B950000000             <1>         mov  ecx, 80
                                <1>       ;mov   cx, 80
000082E8 BF[8E8B0100]           <1>       mov   edi, CommandBuffer
000082ED 30C0                   <1>       xor   al, al
000082EF F3AA                   <1>       rep   stosb
                                <1>       ;cmp   byte [Program_Exit], 0
                                <1>       ;ja    short loc_terminate_trdos
                                <1>
                                <1>       ; 16/01/2016
000082F1 803D[DA6F0000]03       <1>       cmp   byte [CRT_MODE], 3 ; 80*25 color
000082F8 741D                   <1>       je    short pass_set_txt_mode
                                <1>
000082FA E86F98FFFF             <1>       call  set_txt_mode ; set vide mode to 03h
```

```
                            <1>       ; 07/01/2017
000082FF 30C0               <1>       xor    al, al
                            <1>
                            <1> loc_check_active_page:
                            <1>       ;xor   al, al
00008301 3805[468A0100]     <1>       cmp    [ACTIVE_PAGE], al ; 0
00008307 0F842CFFFFFF       <1>         je      loc_TRDOS_prompt
                            <1>       ; AL = 0 = video page 0
0000830D E8A59CFFFF         <1>       call   set_active_page
00008312 E922FFFFFF         <1>         jmp     loc_TRDOS_prompt ; infinitive loop
                            <1>
                            <1> pass_set_txt_mode:
00008317 BE[434D0100]       <1>       mov    esi, nextline
0000831C E844F2FFFF         <1>       call   print_msg
00008321 EBDE               <1>       jmp     short loc_check_active_page
                            <1>
                            <1> rw_char:
                            <1>       ; 13/05/2016
                            <1>       ; 30/01/2016
                            <1>       ; 29/01/2016
                            <1>       ; 17/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 2004-2005
                            <1>
                            <1>       ; DH = cursor row, DL = cursor column
                            <1>       ; BH = 0 = video page number (active page)
                            <1>
                            <1>       ;xor   bh, bh ; 0 = video page 0
                            <1>
                            <1> readnextchar:
00008323 30E4               <1>       xor     ah, ah
00008325 E8C38BFFFF         <1>       call   int16h
0000832A 20C0               <1>       and    al, al
0000832C 7432               <1>       jz     short loc_arrow
0000832E 3CE0               <1>       cmp    al, 0E0h
00008330 742E               <1>       je     short loc_arrow
00008332 3C08               <1>       cmp    al, 08h
00008334 7542               <1>       jne    short char_return
                            <1> loc_back:
00008336 3A15[3E8B0100]     <1>       cmp    dl, [CursorColumn]
0000833C 76E5               <1>       jna     short readnextchar
                            <1> prev_column:
0000833E FECA               <1>       dec    dl
                            <1> set_cursor_pos:
                            <1>       ;push  dx
00008340 52                 <1>       push   edx ; 29/12/2017
                            <1>       ;xor   bh, bh ; 0 = video page 0
                            <1>       ; DH = Row, DL = Column
00008341 E859A0FFFF         <1>       call   _set_cpos ; 17/01/2016
00008346 5A                 <1>         pop  edx ; 29/12/2017
                            <1>       ;pop   dx
                            <1>       ;movzx ebx, dl
00008347 88D3               <1>       mov    bl, dl
00008349 2A1D[3E8B0100]     <1>       sub    bl, [CursorColumn]
0000834F B020               <1>       mov    al, 20h
00008351 8883[8E8B0100]     <1>       mov    [CommandBuffer+ebx], al
                            <1>       ;sub   bh, bh ; video page 0
                            <1>       ;mov   cx, 1
00008357 B307               <1>       mov    bl, 7 ; color attribute
00008359 E8209FFFFF         <1>       call   _write_c_current ; 17/01/2016
                            <1>       ;mov   dx, [CURSOR_POSN]
0000835E EBC3               <1>       jmp    short readnextchar
                            <1> loc_arrow:
00008360 80FC4B             <1>       cmp    ah, 4Bh
00008363 74D1               <1>       je     short loc_back
00008365 80FC53             <1>       cmp    ah, 53h
00008368 74CC               <1>       je      short loc_back
0000836A 80FC4D             <1>       cmp    ah, 4Dh
0000836D 75B4               <1>       jne    short readnextchar
0000836F 80FA4F             <1>       cmp    dl, 79
00008372 73AF               <1>       jnb    short readnextchar
00008374 FEC2               <1>       inc    dl
00008376 EBC8               <1>       jmp    short set_cursor_pos
                            <1> char_return:
00008378 0FB6DA             <1>       movzx  ebx, dl
0000837B 2A1D[3E8B0100]     <1>       sub    bl, [CursorColumn]
00008381 3C20               <1>       cmp    al, 20h
00008383 721D               <1>       jb     short loc_escape
00008385 8883[8E8B0100]     <1>       mov    [CommandBuffer+ebx], al
0000838B 80FA4F             <1>       cmp    dl, 79
0000838E 7393               <1>       jnb    short readnextchar
00008390 66BB0700           <1>       mov    bx, 7 ; color attribute
00008394 E86C9FFFFF         <1>       call   _write_tty
00008399 668B15[368A0100]   <1>       mov    dx, [CURSOR_POSN] ; video page 0
000083A0 EB81               <1>         jmp     readnextchar
                            <1> loc_escape:
000083A2 3C1B               <1>       cmp    al, 1Bh
000083A4 7418               <1>       je     short rw_char_retn
                            <1>       ;
000083A6 3C0D               <1>       cmp    al, 0Dh ; CR
000083A8 0F8575FFFFFF       <1>         jne     readnextchar
                            <1>       ; 13/05/2016
000083AE 66BB0700           <1>       mov    bx, 7 ; attribute/color (bl)
                            <1>                    ; video page 0 (bh=0)
000083B2 E84E9FFFFF         <1>       call   _write_tty
                            <1>       ;mov   bx, 7  ; attribute/color
                            <1>                     ; video page 0 (bh=0)
000083B7 B00A               <1>       mov    al, 0Ah ; LF
000083B9 E8479FFFFF         <1>       call   _write_tty
                            <1> rw_char_retn:
000083BE C3                 <1>       retn
                            <1>
                            <1> show_date:
                            <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>         ; 2004-2005
                            <1>
```

```
                        <1>       ;mov   ah, 04h
                        <1>       ;call  int1Ah
000083BF E85FE7FFFF     <1>       call   RTC_40 ; GET RTC DATE
                        <1>
000083C4 88D0           <1>       mov    al, dl
000083C6 E8158BFFFF     <1>       call   bcd_to_ascii
000083CB 66A3[E5410100] <1>       mov    [Day], ax
                        <1>
000083D1 88F0           <1>       mov    al, dh
000083D3 E8088BFFFF     <1>       call   bcd_to_ascii
000083D8 66A3[E8410100] <1>       mov    [Month], ax
                        <1>
000083DE 88E8           <1>       mov    al, ch
000083E0 E8FB8AFFFF     <1>       call   bcd_to_ascii
000083E5 66A3[EB410100] <1>       mov    [Century], ax
                        <1>
000083EB 88C8           <1>       mov    al, cl
000083ED E8EE8AFFFF     <1>       call   bcd_to_ascii
000083F2 66A3[ED410100] <1>       mov    word [Year], ax
                        <1>
000083F8 BE[D5410100]   <1>       mov    esi, Msg_Show_Date
000083FD E863F1FFFF     <1>       call   print_msg
                        <1>
00008402 C3             <1>       retn
                        <1>
                        <1> set_date:
                        <1>       ; 13/05/2016
                        <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>         ; 2004-2005
                        <1>
00008403 BE[B9410100]   <1>       mov    esi, Msg_Enter_Date
00008408 E858F1FFFF     <1>       call   print_msg
                        <1>
                        <1> loc_enter_day_1:
0000840D 30E4           <1>       xor     ah, ah
0000840F E8D98AFFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
00008414 3C0D           <1>       cmp    al, 13
00008416 0F84B7010000   <1>       je     loc_set_date_retn
0000841C 3C1B           <1>       cmp    al, 27
0000841E 0F84AF010000   <1>       je     loc_set_date_retn
00008424 A2[E5410100]   <1>       mov    [Day], al
00008429 3C30           <1>       cmp    al, '0'
0000842B 0F82AD010000   <1>       jb     loc_set_date_stc_0
00008431 3C33           <1>       cmp    al, '3'
00008433 0F87A5010000   <1>       ja     loc_set_date_stc_0
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
00008439 B307           <1>       mov    bl, 7
0000843B E8C59EFFFF     <1>       call   _write_tty
                        <1> loc_enter_day_2:
00008440 30E4           <1>       xor     ah, ah
00008442 E8A68AFFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
00008447 3C1B           <1>       cmp    al, 27
00008449 0F8484010000   <1>         je      loc_set_date_retn
0000844F A2[E6410100]   <1>       mov    [Day+1], al
00008454 3C30           <1>       cmp    al, '0'
00008456 0F828C010000   <1>         jb      loc_set_date_stc_1
0000845C 3C39           <1>       cmp    al, '9'
0000845E 0F8784010000   <1>         ja      loc_set_date_stc_1
00008464 803D[E5410100]33 <1>     cmp    byte [Day], '3'
0000846B 7208           <1>       jb     short pass_set_day_31
0000846D 3C31           <1>       cmp    al, '1'
0000846F 0F8773010000   <1>         ja      loc_set_date_stc_1
                        <1> pass_set_day_31:
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
00008475 B307           <1>       mov    bl, 7
00008477 E8899EFFFF     <1>       call   _write_tty
                        <1> loc_enter_separator_1:
0000847C 28E4           <1>       sub     ah, ah ; 0
0000847E E86A8AFFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
00008483 3C1B           <1>       cmp    al, 27
00008485 0F8448010000   <1>         je      loc_set_date_retn
0000848B 3C2D           <1>       cmp    al, '-'
0000848D 7408           <1>       je     short pass_set_date_separator_1
0000848F 3C2F           <1>       cmp    al, '/'
00008491 0F856C010000   <1>         jne     loc_set_date_stc_2
                        <1> pass_set_date_separator_1:
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
00008497 B307           <1>       mov    bl, 7
00008499 E8679EFFFF     <1>       call   _write_tty
                        <1> loc_enter_month_1:
0000849E 30E4           <1>       xor     ah, ah ; 0
000084A0 E8488AFFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
000084A5 3C1B           <1>       cmp    al, 27
000084A7 0F8426010000   <1>         je      loc_set_date_retn
000084AD A2[E8410100]   <1>       mov    [Month], al
000084B2 3C30           <1>       cmp    al, '0'
000084B4 0F8264010000   <1>         jb      loc_set_date_stc_3
000084BA 3C31           <1>       cmp    al, '1'
000084BC 0F875C010000   <1>         ja      loc_set_date_stc_3
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
000084C2 B307           <1>       mov    bl, 7
000084C4 E83C9EFFFF     <1>       call   _write_tty
                        <1> loc_enter_month_2:
```

```
000084C9 30E4                <1>       xor     ah, ah
000084CB E81D8AFFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
000084D0 3C1B                <1>       cmp    al, 27
000084D2 0F84FB000000        <1>         je      loc_set_date_retn
000084D8 A2[E9410100]        <1>       mov    [Month+1], al
000084DD 3C30                <1>       cmp    al, '0'
000084DF 0F8254010000        <1>         jb      loc_set_date_stc_4
000084E5 3C39                <1>       cmp    al, '9'
000084E7 0F874C010000        <1>         ja      loc_set_date_stc_4
000084ED 803D[E8410100]31    <1>       cmp    byte [Month], '1'
000084F4 7208                <1>       jb     short pass_set_month_12
000084F6 3C32                <1>       cmp    al, '2'
000084F8 0F873B010000        <1>         ja      loc_set_date_stc_4
                             <1> pass_set_month_12:
                             <1>       ; 13/05/2016
                             <1>       ;mov   bx, 7 ; attribute/color (bl)
                             <1>                    ; video page 0 (bh)
000084FE B307                <1>       mov    bl, 7
00008500 E8009EFFFF          <1>       call   _write_tty
                             <1> loc_enter_separator_2:
00008505 28E4                <1>       sub     ah, ah
00008507 E8E189FFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
0000850C 3C1B                <1>       cmp    al, 27
0000850E 0F84BF000000        <1>         je      loc_set_date_retn
00008514 3C2D                <1>       cmp    al, '-'
00008516 7408                <1>       je     short pass_set_date_separator_2
00008518 3C2F                <1>       cmp    al, '/'
0000851A 0F8534010000        <1>         jne     loc_set_date_stc_5
                             <1> pass_set_date_separator_2:
                             <1>       ; 13/05/2016
                             <1>       ;mov   bx, 7 ; attribute/color (bl)
                             <1>                    ; video page 0 (bh)
00008520 B307                <1>       mov    bl, 7
00008522 E8DE9DFFFF          <1>       call   _write_tty
                             <1> loc_enter_year_1:
00008527 30E4                <1>       xor    ah, ah
00008529 E8BF89FFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
0000852E 3C1B                <1>       cmp    al, 27
00008530 0F849D000000        <1>         je      loc_set_date_retn
00008536 A2[ED410100]        <1>       mov    [Year], al
0000853B 3C30                <1>       cmp    al, '0'
0000853D 0F822C010000        <1>         jb      loc_set_date_stc_6
00008543 3C39                <1>       cmp    al, '9'
00008545 0F8724010000        <1>         ja      loc_set_date_stc_6
                             <1>       ; 13/05/2016
                             <1>       ;mov   bx, 7 ; attribute/color (bl)
                             <1>                    ; video page 0 (bh)
0000854B B307                <1>       mov    bl, 7
0000854D E8B39DFFFF          <1>       call   _write_tty
                             <1> loc_enter_year_2:
00008552 30E4                <1>       xor    ah, ah
00008554 E89489FFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
00008559 3C1B                <1>       cmp    al, 27
0000855B 7476                <1>       je     short loc_set_date_retn
0000855D A2[EE410100]        <1>       mov    byte [Year+1], al
00008562 3C30                <1>       cmp    al, '0'
00008564 0F8220010000        <1>         jb      loc_set_date_stc_7
0000856A 3C39                <1>       cmp    al, '9'
0000856C 0F8718010000        <1>         ja      loc_set_date_stc_7
                             <1>       ; 13/05/2016
                             <1>       ;mov   bx, 7 ; attribute/color (bl)
                             <1>                    ; video page 0 (bh)
00008572 B307                <1>       mov    bl, 7
00008574 E88C9DFFFF          <1>       call   _write_tty
                             <1> loc_set_date_get_lchar_again:
00008579 28E4                <1>       sub    ah, ah ; 0
0000857B E86D89FFFF          <1>       call   int16h
                             <1>       ; AL = ASCII Code of the Character
00008580 3C0D                <1>       cmp    al, 13 ; ENTER key
00008582 7412                <1>       je     short loc_set_date_progress
00008584 3C1B                <1>       cmp    al, 27 ; ESC key
00008586 744B                <1>       je     short loc_set_date_retn
                             <1>       ;
00008588 E82A010000          <1>       call   check_for_backspace
0000858D 75EA                <1>       jne    short loc_set_date_get_lchar_again
                             <1>
                             <1> loc_set_date_bs_8:
0000858F E811010000          <1>       call   write_backspace
00008594 EBBC                <1>       jmp    short loc_enter_year_2
                             <1>
                             <1> loc_set_date_progress:
                             <1>       ; Get Current Date
                             <1>       ;mov   ah, 04h
                             <1>       ;call  int1Ah
00008596 E888E5FFFF          <1>       call   RTC_40 ; GET RTC DATE
                             <1>       ; CH = century (in BCD)
                             <1>
0000859B 66A1[ED410100]      <1>       mov    ax, [Year]
000085A1 662D3030            <1>       sub    ax, '00'
000085A5 C0E004              <1>       shl    al, 4 ; * 16
000085A8 88C1                <1>       mov    cl, al
000085AA 00E1                <1>       add    cl, ah
000085AC 66A1[E8410100]      <1>       mov    ax, [Month]
000085B2 662D3030            <1>       sub    ax, '00'
000085B6 C0E004              <1>       shl    al, 4 ; * 16
000085B9 88C6                <1>       mov    dh, al
000085BB 00E6                <1>       add    dh, ah
000085BD 66A1[E5410100]      <1>       mov    ax, [Day]
000085C3 662D3030            <1>       sub    ax, '00'
000085C7 C0E004              <1>       shl    al, 4 ; * 16
000085CA 88C2                <1>       mov    dl, al
```

```
000085CC 00E2                <1>       add    dl, ah
                             <1>
                             <1>       ;mov   ah, 05h
                             <1>       ;call  int1Ah
000085CE E87DE5FFFF          <1>       call   RTC_50 ; SET RTC DATE
                             <1>
                             <1> loc_set_date_retn:
000085D3 BE[434D0100]        <1>       mov    esi, nextline
000085D8 E888EFFFFF          <1>       call   print_msg
000085DD C3                  <1>       retn
                             <1>
                             <1> loc_set_date_stc_0:
                             <1>       ;xor   bh, bh ; video page 0
000085DE E80E9EFFFF          <1>       call   beeper ; BEEP !
000085E3 E925FEFFFF          <1>         jmp     loc_enter_day_1
                             <1> loc_set_date_stc_1:
000085E8 E8CA000000          <1>       call   check_for_backspace
000085ED 740A                <1>       je     short loc_set_date_bs_1
                             <1>       ;xor   bh, bh ; video page 0
000085EF E8FD9DFFFF          <1>       call   beeper ; BEEP !
000085F4 E947FEFFFF          <1>         jmp     loc_enter_day_2
                             <1> loc_set_date_bs_1:
000085F9 E8A7000000          <1>       call   write_backspace
000085FE E90AFEFFFF          <1>         jmp     loc_enter_day_1
                             <1> loc_set_date_stc_2:
00008603 E8AF000000          <1>       call   check_for_backspace
00008608 740A                <1>       je     short loc_set_date_bs_2
                             <1>       ;xor   bh, bh ; video page 0
0000860A E8E29DFFFF          <1>       call   beeper ; BEEP !
0000860F E968FEFFFF          <1>         jmp     loc_enter_separator_1
                             <1> loc_set_date_bs_2:
00008614 E88C000000          <1>       call   write_backspace
00008619 E922FEFFFF          <1>         jmp     loc_enter_day_2
                             <1> loc_set_date_stc_3:
0000861E E894000000          <1>       call   check_for_backspace
00008623 740A                <1>       je short loc_set_date_bs_3
                             <1>       ;xor   bh, bh ; video page 0
00008625 E8C79DFFFF          <1>       call   beeper ; BEEP !
0000862A E96FFEFFFF          <1>         jmp     loc_enter_month_1
                             <1> loc_set_date_bs_3:
0000862F E871000000          <1>       call   write_backspace
00008634 E943FEFFFF          <1>         jmp     loc_enter_separator_1
                             <1> loc_set_date_stc_4:
00008639 E879000000          <1>       call   check_for_backspace
0000863E 740A                <1>       je     short loc_set_date_bs_4
                             <1>       ;xor   bh, bh ; video page 0
00008640 E8AC9DFFFF          <1>       call   beeper ; BEEP !
00008645 E97FFEFFFF          <1>         jmp     loc_enter_month_2
                             <1> loc_set_date_bs_4:
0000864A E856000000          <1>       call   write_backspace
0000864F E94AFEFFFF          <1>         jmp     loc_enter_month_1
                             <1> loc_set_date_stc_5:
00008654 E85E000000          <1>       call   check_for_backspace
00008659 740A                <1>       je     short loc_set_date_bs_5
                             <1>       ;xor   bh, bh ; video page 0
0000865B E8919DFFFF          <1>       call   beeper ; BEEP !
00008660 E9A0FEFFFF          <1>         jmp     loc_enter_separator_2
                             <1> loc_set_date_bs_5:
00008665 E83B000000          <1>       call   write_backspace
0000866A E95AFEFFFF          <1>         jmp     loc_enter_month_2
                             <1> loc_set_date_stc_6:
0000866F E843000000          <1>       call   check_for_backspace
00008674 740A                <1>         je      short  loc_set_date_bs_6
                             <1>       ;xor   bh, bh ; video page 0
00008676 E8769DFFFF          <1>       call   beeper ; BEEP !
0000867B E9A7FEFFFF          <1>         jmp     loc_enter_year_1
                             <1> loc_set_date_bs_6:
00008680 E820000000          <1>       call   write_backspace
00008685 E97BFEFFFF          <1>         jmp     loc_enter_separator_2
                             <1> loc_set_date_stc_7:
0000868A E828000000          <1>       call   check_for_backspace
0000868F 740A                <1>       je     short loc_set_date_bs_7
                             <1>       ;xor   bh, bh ; video page 0
00008691 E85B9DFFFF          <1>       call   beeper ; BEEP !
00008696 E9B7FEFFFF          <1>         jmp     loc_enter_year_2
                             <1> loc_set_date_bs_7:
0000869B E805000000          <1>       call   write_backspace
000086A0 E982FEFFFF          <1>         jmp     loc_enter_year_1
                             <1>
                             <1> write_backspace:
                             <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
000086A5 B008                <1>       mov    al, 08h ; BACKSPACE
                             <1>       ; 13/05/2016
000086A7 66BB0700            <1>       mov    bx, 7 ; bl = attribute/color
                             <1>                    ; bh = video page = 0
000086AB E8559CFFFF          <1>       call   _write_tty
000086B0 B020                <1>       mov    al, 20h ; BLANK/SPACE char
                             <1>       ;mov   bx, 7 ; attribute/color
                             <1>       ;call  _write_c_current
                             <1>       ;retn
000086B2 E9C79BFFFF          <1>       jmp    _write_c_current
                             <1>
                             <1> check_for_backspace:
                             <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
000086B7 663D080E            <1>       cmp    ax, 0E08h
000086BB 7410                <1>       je     short cfbs_retn
000086BD 663DE04B            <1>       cmp    ax, 4BE0h
000086C1 740A                <1>       je     short cfbs_retn
000086C3 663D004B            <1>       cmp    ax, 4B00h
000086C7 7404                <1>       je     short cfbs_retn
000086C9 663DE053            <1>       cmp    ax, 53E0h
                             <1> cfbs_retn:
000086CD 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
000086CE E8DFE3FFFF     <1>       call   RTC_20 ; GET RTC TIME
                        <1>
000086D3 88E8           <1>       mov    al, ch
000086D5 E80688FFFF     <1>       call   bcd_to_ascii
000086DA 66A3[13420100] <1>       mov    [Hour], ax
                        <1>
000086E0 88C8           <1>       mov    al, cl
000086E2 E8F987FFFF     <1>       call   bcd_to_ascii
000086E7 66A3[16420100] <1>       mov    [Minute], ax
                        <1>
000086ED 88F0           <1>       mov    al, dh
000086EF E8EC87FFFF     <1>       call   bcd_to_ascii
000086F4 66A3[19420100] <1>       mov    [Second], ax
                        <1>
000086FA BE[03420100]   <1>       mov    esi, Msg_Show_Time
000086FF E861EEFFFF     <1>       call   print_msg
00008704 C3             <1>       retn
                        <1>
                        <1> set_time:
                        <1>       ; 13/05/2016
                        <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>         ; 2004-2005
                        <1>
00008705 BE[F2410100]   <1>       mov    esi, Msg_Enter_Time
0000870A E856EEFFFF     <1>       call   print_msg
                        <1>
                        <1> loc_enter_hour_1:
0000870F 30E4           <1>       xor     ah, ah
00008711 E8D787FFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
00008716 3C0D           <1>       cmp    al, 13 ; ENTER key
00008718 0F84AE010000   <1>         je      loc_set_time_retn
0000871E 3C1B           <1>       cmp    al, 27 ; ESC key
00008720 0F84A6010000   <1>         je      loc_set_time_retn
00008726 A2[13420100]   <1>       mov    [Hour], al
0000872B 3C30           <1>       cmp    al, '0'
0000872D 0F82A4010000   <1>         jb      loc_set_time_stc_0
00008733 3C32           <1>       cmp    al, '2'
00008735 0F879C010000   <1>         ja      loc_set_time_stc_0
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
0000873B B307           <1>       mov    bl, 7
0000873D E8C39BFFFF     <1>       call   _write_tty
                        <1> loc_enter_hour_2:
00008742 30E4           <1>       xor     ah, ah
00008744 E8A487FFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
00008749 3C1B           <1>       cmp    al, 27
0000874B 0F847B010000   <1>         je      loc_set_time_retn
00008751 A2[14420100]   <1>       mov    [Hour+1], al
00008756 3C30           <1>       cmp    al, '0'
00008758 0F8283010000   <1>         jb      loc_set_time_stc_1
0000875E 3C39           <1>       cmp    al, '9'
00008760 0F877B010000   <1>       ja     loc_set_time_stc_1
00008766 803D[13420100]32 <1>       cmp     byte [Hour], '2'
0000876D 7208           <1>       jb     short pass_set_time_24
0000876F 3C34           <1>       cmp    al, '4'
00008771 0F876A010000   <1>         ja      loc_set_time_stc_1
                        <1> pass_set_time_24:
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
00008777 B307           <1>       mov    bl, 7
00008779 E8879BFFFF     <1>       call   _write_tty
                        <1> loc_enter_time_separator_1:
0000877E 28E4           <1>       sub     ah, ah ; 0
00008780 E86887FFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
00008785 3C1B           <1>       cmp    al, 27
00008787 0F843F010000   <1>         je      loc_set_time_retn
0000878D 3C3A           <1>       cmp    al, ':'
0000878F 0F8567010000   <1>         jne     loc_set_time_stc_2
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
00008795 B307           <1>       mov    bl, 7
00008797 E8699BFFFF     <1>       call   _write_tty
                        <1> loc_enter_minute_1:
0000879C 30E4           <1>       xor     ah, ah
0000879E E84A87FFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
000087A3 3C1B           <1>       cmp    al, 27
000087A5 0F8421010000   <1>         je      loc_set_time_retn
000087AB A2[16420100]   <1>       mov    [Minute], al
000087B0 3C30           <1>       cmp    al, '0'
000087B2 0F825F010000   <1>         jb      loc_set_time_stc_3
000087B8 3C35           <1>       cmp    al, '5'
000087BA 0F8757010000   <1>         ja      loc_set_time_stc_3
                        <1>       ; 13/05/2016
                        <1>       ;mov   bx, 7 ; attribute/color (bl)
                        <1>                    ; video page 0 (bh)
000087C0 B307           <1>       mov    bl, 7
000087C2 E83E9BFFFF     <1>       call   _write_tty
                        <1> loc_enter_minute_2:
000087C7 30E4           <1>       xor     ah, ah
000087C9 E81F87FFFF     <1>       call   int16h
                        <1>       ; AL = ASCII Code of the Character
000087CE 3C1B           <1>       cmp    al, 27
000087D0 0F84F6000000   <1>         je      loc_set_time_retn
```

```
891 000087D6 A2[17420100]        <1>       mov    [Minute+1], al
892 000087DB 3C30                <1>       cmp    al, '0'
893 000087DD 0F824F010000        <1>         jb      loc_set_time_stc_4
894 000087E3 3C39                <1>       cmp    al, '9'
895 000087E5 0F8747010000        <1>         ja      loc_set_time_stc_4
896                              <1>       ; 13/05/2016
897                              <1>       ;mov   bx, 7 ; attribute/color (bl)
898                              <1>                    ; video page 0 (bh)
899 000087EB B307                <1>       mov    bl, 7
900 000087ED E8139BFFFF          <1>       call   _write_tty
901                              <1> loc_enter_time_separator_2:
902 000087F2 66C705[19420100]30- <1>       mov    word [Second], 3030h
902 000087FA 30                  <1>
903 000087FB 28E4                <1>       sub     ah, ah
904 000087FD E8EB86FFFF          <1>       call   int16h
905                              <1>       ; AL = ASCII Code of the Character
906 00008802 3C0D                <1>       cmp    al, 13
907 00008804 0F8485000000        <1>         je      loc_set_time_progress
908 0000880A 3C1B                <1>       cmp    al, 27
909 0000880C 0F84BA000000        <1>         je      loc_set_time_retn
910 00008812 3C3A                <1>       cmp    al, ':'
911 00008814 0F8533010000        <1>         jne     loc_set_time_stc_5
912                              <1>       ; 13/05/2016
913                              <1>       ;mov   bx, 7 ; attribute/color (bl)
914                              <1>                    ; video page 0 (bh)
915 0000881A B307                <1>       mov    bl, 7
916 0000881C E8E49AFFFF          <1>       call   _write_tty
917                              <1> loc_enter_second_1:
918 00008821 30E4                <1>       xor     ah, ah
919 00008823 E8C586FFFF          <1>       call   int16h
920                              <1>       ; AL = ASCII Code of the Character
921 00008828 3C0D                <1>       cmp    al, 13
922 0000882A 7463                <1>       je     short loc_set_time_progress
923 0000882C 3C1B                <1>       cmp    al, 27
924 0000882E 0F8498000000        <1>         je      loc_set_time_retn
925 00008834 A2[19420100]        <1>       mov    [Second], al
926 00008839 3C30                <1>       cmp    al, '0'
927 0000883B 0F8227010000        <1>         jb      loc_set_time_stc_6
928 00008841 3C35                <1>       cmp    al, '5'
929 00008843 0F871F010000        <1>         ja      loc_set_time_stc_6
930                              <1>       ; 13/05/2016
931                              <1>       ;mov   bx, 7 ; attribute/color (bl)
932                              <1>                    ; video page 0 (bh)
933 00008849 B307                <1>       mov    bl, 7
934 0000884B E8B59AFFFF          <1>       call   _write_tty
935                              <1> loc_enter_second_2:
936 00008850 30E4                <1>       xor     ah, ah
937 00008852 E89686FFFF          <1>       call   int16h
938                              <1>       ; AL = ASCII Code of the Character
939 00008857 3C1B                <1>       cmp    al, 27
940 00008859 7471                <1>       je     short loc_set_time_retn
941 0000885B 3C30                <1>       cmp    al, '0'
942 0000885D 0F8229010000        <1>         jb      loc_set_time_stc_7
943 00008863 3C39                <1>       cmp    al, '9'
944 00008865 0F8721010000        <1>         ja      loc_set_time_stc_7
945                              <1>       ; 13/05/2016
946                              <1>       ;mov   bx, 7 ; attribute/color (bl)
947                              <1>                    ; video page 0 (bh)
948 0000886B B307                <1>       mov    bl, 7
949 0000886D E8939AFFFF          <1>       call   _write_tty
950                              <1> loc_set_time_get_lchar_again:
951 00008872 28E4                <1>       sub    ah, ah ; 0
952 00008874 E87486FFFF          <1>       call   int16h
953                              <1>       ; AL = ASCII Code of the Character
954 00008879 3C0D                <1>       cmp    al, 13
955 0000887B 7412                <1>       je     short loc_set_time_progress
956 0000887D 3C1B                <1>       cmp    al, 27
957 0000887F 744B                <1>       je     short loc_set_time_retn
958                              <1>       ;
959 00008881 E831FEFFFF          <1>       call   check_for_backspace
960 00008886 75EA                <1>       jne    short loc_set_time_get_lchar_again
961                              <1>
962                              <1> loc_set_time_bs_8:
963 00008888 E818FEFFFF          <1>       call   write_backspace
964 0000888D EBC1                <1>       jmp    short loc_enter_second_2
965                              <1>
966                              <1> loc_set_time_progress:
967                              <1>       ; Get Current Time
968                              <1>       ;mov   ah, 02h
969                              <1>       ;call  int1Ah
970 0000888F E81EE2FFFF          <1>       call   RTC_20 ; GET RTC TIME
971                              <1>       ;DL = Daylight Savings Enable option (0-1)
972                              <1>
973 00008894 66A1[13420100]      <1>       mov    ax, [Hour]
974 0000889A 662D3030            <1>       sub    ax, '00'
975 0000889E C0E004              <1>       shl    al, 4 ; * 16
976 000088A1 88C5                <1>       mov    ch, al
977 000088A3 00E5                <1>       add    ch, ah
978 000088A5 66A1[16420100]      <1>       mov    ax, [Minute]
979 000088AB 662D3030            <1>       sub    ax, '00'
980 000088AF C0E004              <1>       shl    al, 4 ; * 16
981 000088B2 88C1                <1>       mov    cl, al
982 000088B4 00E1                <1>       add    cl, ah
983 000088B6 66A1[19420100]      <1>       mov    ax, [Second]
984 000088BC 662D3030            <1>       sub    ax, '00'
985 000088C0 C0E004              <1>       shl    al, 4 ; * 16
986 000088C3 88C6                <1>       mov    dh, al
987 000088C5 00E6                <1>       add    dh, ah
988                              <1>
989                              <1>       ;mov   ah, 03h
990                              <1>       ;call  int1Ah
991 000088C7 E815E2FFFF          <1>       call   RTC_30 ; SET RTC TIME
992                              <1>
993                              <1> loc_set_time_retn:
994 000088CC BE[434D0100]        <1>       mov    esi, nextline
```

```
 995 000088D1 E88FECFFFF          <1>       call   print_msg
 996 000088D6 C3                  <1>       retn
 997                              <1>
 998                              <1> loc_set_time_stc_0:
 999                              <1>       ;xor   bh, bh ; video page 0
1000 000088D7 E8159BFFFF          <1>       call   beeper ; BEEP !
1001 000088DC E92EFEFFFF          <1>         jmp     loc_enter_hour_1
1002                              <1> loc_set_time_stc_1:
1003 000088E1 E8D1FDFFFF          <1>       call   check_for_backspace
1004 000088E6 740A                <1>       je     short loc_set_time_bs_1
1005                              <1>       ;xor   bh, bh ; video page 0
1006 000088E8 E8049BFFFF          <1>       call   beeper ; BEEP !
1007 000088ED E950FEFFFF          <1>         jmp     loc_enter_hour_2
1008                              <1> loc_set_time_bs_1:
1009 000088F2 E8AEFDFFFF          <1>       call   write_backspace
1010 000088F7 E913FEFFFF          <1>         jmp     loc_enter_hour_1
1011                              <1> loc_set_time_stc_2:
1012 000088FC E8B6FDFFFF          <1>       call   check_for_backspace
1013 00008901 740A                <1>       je     short loc_set_time_bs_2
1014                              <1>       ;xor   bh, bh ; video page 0
1015 00008903 E8E99AFFFF          <1>       call   beeper ; BEEP !
1016 00008908 E971FEFFFF          <1>         jmp     loc_enter_time_separator_1
1017                              <1> loc_set_time_bs_2:
1018 0000890D E893FDFFFF          <1>       call   write_backspace
1019 00008912 E92BFEFFFF          <1>         jmp     loc_enter_hour_2
1020                              <1> loc_set_time_stc_3:
1021 00008917 E89BFDFFFF          <1>       call   check_for_backspace
1022 0000891C 740A                <1>       je     short loc_set_time_bs_3
1023                              <1>       ;xor   bh, bh ; video page 0
1024 0000891E E8CE9AFFFF          <1>       call   beeper ; BEEP !6
1025 00008923 E974FEFFFF          <1>         jmp     loc_enter_minute_1
1026                              <1> loc_set_time_bs_3:
1027 00008928 E878FDFFFF          <1>       call   write_backspace
1028 0000892D E94CFEFFFF          <1>         jmp     loc_enter_time_separator_1
1029                              <1> loc_set_time_stc_4:
1030 00008932 E880FDFFFF          <1>       call   check_for_backspace
1031 00008937 740A                <1>       je     short loc_set_time_bs_4
1032                              <1>       ;xor   bh, bh ; video page 0
1033 00008939 E8B39AFFFF          <1>       call   beeper ; BEEP !
1034 0000893E E984FEFFFF          <1>         jmp     loc_enter_minute_2
1035                              <1> loc_set_time_bs_4:
1036 00008943 E85DFDFFFF          <1>       call   write_backspace
1037 00008948 E94FFEFFFF          <1>         jmp     loc_enter_minute_1
1038                              <1> loc_set_time_stc_5:
1039 0000894D E865FDFFFF          <1>       call   check_for_backspace
1040 00008952 740A                <1>       je     short loc_set_time_bs_5
1041                              <1>       ;xor   bh, bh ; video page 0
1042 00008954 E8989AFFFF          <1>       call   beeper ; BEEP !
1043 00008959 E994FEFFFF          <1>         jmp     loc_enter_time_separator_2
1044                              <1> loc_set_time_bs_5:
1045 0000895E E842FDFFFF          <1>       call   write_backspace
1046 00008963 E95FFEFFFF          <1>         jmp     loc_enter_minute_2
1047                              <1> loc_set_time_stc_6:
1048 00008968 E84AFDFFFF          <1>       call   check_for_backspace
1049 0000896D 7413                <1>       je     short loc_set_time_bs_6
1050                              <1>       ;xor   bh, bh ; video page 0
1051 0000896F E87D9AFFFF          <1>       call   beeper ; BEEP !
1052 00008974 66C705[19420100]30- <1>       mov    word [Second], 3030h
1052 0000897C 30                  <1>
1053 0000897D E99FFEFFFF          <1>         jmp     loc_enter_second_1
1054                              <1> loc_set_time_bs_6:
1055 00008982 E81EFDFFFF          <1>       call   write_backspace
1056 00008987 E966FEFFFF          <1>         jmp     loc_enter_time_separator_2
1057                              <1> loc_set_time_stc_7:
1058 0000898C E826FDFFFF          <1>       call   check_for_backspace
1059 00008991 740A                <1>       je     short loc_set_time_bs_7
1060                              <1>       ;xor   bh, bh ; video page 0
1061 00008993 E8599AFFFF          <1>       call   beeper ; BEEP !
1062 00008998 E9B3FEFFFF          <1>         jmp     loc_enter_second_2
1063                              <1> loc_set_time_bs_7:
1064 0000899D E803FDFFFF          <1>       call   write_backspace
1065 000089A2 E97AFEFFFF          <1>         jmp     loc_enter_second_1
1066                              <1>
1067                              <1> print_volume_info:
1068                              <1>       ; 01/03/2016
1069                              <1>       ; 08/02/2016
1070                              <1>       ; 06/02/2016
1071                              <1>       ; 04/02/2016
1072                              <1>       ; 18/01/2016 (TRDOS 386 = TRDOS v2.0)
1073                              <1>       ; 25/10/2009
1074                              <1>       ;
1075                              <1>       ; "Volume Serial No: "
1076                              <1>       ;
1077                              <1>       ; INPUT  : AL = DOS Drive Number
1078                              <1>       ; OUTPUT : AH = FS Type
1079                              <1>       ;          AL = DOS Drive Name
1080                              <1>       ; CF = 0 -> OK
1081                              <1>       ; CF = 1 -> Drive not ready
1082                              <1>
1083 000089A7 88C4                <1>       mov    ah, al
1084 000089A9 28C0                <1>       sub    al, al
1085 000089AB 0FB7F0              <1>       movzx  esi, ax
1086 000089AE 81C600010900        <1>       add    esi, Logical_DOSDisks
1087 000089B4 8A06                <1>       mov    al, [esi]
1088 000089B6 3C41                <1>       cmp    al, 'A'
1089 000089B8 7304                <1>       jnb    short loc_pvi_set_vol_name
1090 000089BA 8A6604              <1>       mov    ah, [esi+LD_FSType]
1091 000089BD C3                  <1>       retn
1092                              <1>
1093                              <1> loc_pvi_set_vol_name:
1094 000089BE A2[4D420100]        <1>       mov    [Vol_Drv_Name], al
1095 000089C3 56                  <1>       push   esi
1096 000089C4 E858010000          <1>       call   move_volume_name_and_serial_no ;;;
1097 000089C9 7302                <1>       jnc    short loc_pvi_mvn_ok
1098 000089CB 5E                  <1>       pop    esi
```

```
1099 000089CC C3                <1>       retn
1100                            <1>
1101                            <1> loc_pvi_mvn_ok:
1102 000089CD 8B3424            <1>       mov    esi, [esp]
1103 000089D0 807E04A1          <1>       cmp    byte [esi+LD_FSType], 0A1h
1104 000089D4 7509              <1>       jne    short loc_pvi_fat_vol_size
1105 000089D6 8B4670            <1>       mov    eax, [esi+LD_FS_VolumeSize]
1106 000089D9 0FB75E11          <1>       movzx  ebx, word [esi+LD_FS_BytesPerSec]
1107 000089DD EB07              <1>       jmp    short loc_vol_size_mul32
1108                            <1> loc_pvi_fat_vol_size:
1109 000089DF 8B4670            <1>       mov    eax, [esi+LD_TotalSectors]
1110 000089E2 0FB75E11          <1>       movzx  ebx, word [esi+LD_BPB+BPB_BytsPerSec]
1111                            <1> loc_vol_size_mul32:
1112 000089E6 F7E3              <1>       mul    ebx
1113 000089E8 09D2              <1>       or     edx, edx
1114 000089EA 7507              <1>       jnz    short loc_vol_size_in_kbytes
1115                            <1> loc_vol_size_in_bytes:
1116 000089EC B9[2B420100]      <1>       mov    ecx, VolSize_Bytes
1117 000089F1 EB0D              <1>       jmp    short loc_write_vol_size_str
1118                            <1> loc_vol_size_in_kbytes:
1119 000089F3 66BB0004          <1>       mov    bx, 1024
1120 000089F7 F7F3              <1>       div    ebx
1121 000089F9 B9[1E420100]      <1>       mov    ecx, VolSize_KiloBytes
1122 000089FE 31D2              <1>       xor    edx, edx ; 0
1123                            <1> loc_write_vol_size_str:
1124 00008A00 890D[13920100]    <1>       mov    [VolSize_Unit1], ecx
1125                            <1>       ;
1126 00008A06 BF[29920100]      <1>       mov    edi, Vol_Tot_Sec_Str_End
1127                            <1>         ;mov byte [edi], 0
1128 00008A0B B90A000000        <1>       mov    ecx, 10
1129                            <1> loc_write_vol_size_chr:
1130 00008A10 F7F1              <1>       div    ecx
1131 00008A12 80C230            <1>       add    dl, '0'
1132 00008A15 4F                <1>       dec    edi
1133 00008A16 8817              <1>       mov    [edi], dl
1134 00008A18 85C0              <1>       test   eax, eax
1135 00008A1A 7404              <1>       jz     short loc_write_vol_size_str_ok
1136 00008A1C 28D2              <1>       sub    dl, dl ; 0
1137 00008A1E EBF0              <1>       jmp    short loc_write_vol_size_chr
1138                            <1>
1139                            <1> loc_write_vol_size_str_ok:
1140 00008A20 893D[1B920100]    <1>       mov    [Vol_Tot_Sec_Str_Start], edi
1141                            <1>       ;
1142 00008A26 BF[36420100]      <1>       mov    edi, Vol_FS_Name
1143 00008A2B 8A4E03            <1>       mov    cl, [esi+LD_FATType]
1144 00008A2E 20C9              <1>       and    cl, cl ; 0 ?
1145 00008A30 7515              <1>       jnz    short loc_write_vol_FAT_str_1
1146 00008A32 66C7075452        <1>       mov    word [edi], 'TR'
1147 00008A37 C7470420465331    <1>       mov    dword [edi+4], ' FS1'
1148                            <1>       ;movzx ebx, word [esi+LD_FS_BytesPerSec]
1149 00008A3E 668B5E11          <1>       mov    bx, [esi+LD_FS_BytesPerSec]
1150 00008A42 8B4674            <1>       mov    eax, [esi+LD_FS_FreeSectors]
1151 00008A45 EB36              <1>       jmp    short loc_vol_freespace_mul32
1152                            <1>
1153                            <1> loc_write_vol_FAT_str_1:
1154 00008A47 66B83332          <1>       mov    ax, '32' ; FAT32
1155 00008A4B 80F902            <1>       cmp    cl, 2 ; [esi+LD_FATType]
1156 00008A4E 7708              <1>       ja     short loc_write_vol_FAT_str_2
1157 00008A50 66B83132          <1>       mov    ax, '12' ; FAT12
1158 00008A54 7202              <1>       jb     short loc_write_vol_FAT_str_2
1159 00008A56 B436              <1>       mov    ah, '6'  ; FAT16
1160                            <1> loc_write_vol_FAT_str_2:
1161 00008A58 C70746415420      <1>       mov    dword [edi], 'FAT '
1162 00008A5E 66894704          <1>       mov    word [edi+4], ax
1163                            <1>       ;
1164                            <1>       ;movzx ebx, word [esi+LD_BPB+BPB_BytsPerSec]
1165 00008A62 668B5E11          <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec]
1166 00008A66 8B4674            <1>       mov    eax, [esi+LD_FreeSectors]
1167                            <1>
1168                            <1> loc_vol_freespace_recalc0:
1169                            <1>       ; 01/03/2016
1170 00008A69 83F8FF            <1>       cmp    eax, 0FFFFFFFFh
1171 00008A6C 720F              <1>       jb     short loc_vol_freespace_mul32
1172                            <1>       ;inc   eax ; 0
1173 00008A6E 20C9              <1>       and    cl, cl ; byte [esi+LD_FATType]
1174 00008A70 740B              <1>       jz     short loc_vol_freespace_mul32
1175 00008A72 53                <1>       push   ebx
1176 00008A73 66BB00FF          <1>       mov    bx, 0FF00h ; recalculate free sectors
1177 00008A77 E876490000        <1>       call   calculate_fat_freespace
1178 00008A7C 5B                <1>       pop    ebx
1179                            <1>
1180                            <1> loc_vol_freespace_mul32:
1181 00008A7D F7E3              <1>       mul    ebx
1182 00008A7F 09D2              <1>       or     edx, edx
1183 00008A81 7507              <1>       jnz    short loc_vol_fspace_in_kbytes
1184                            <1> loc_vol_fspace_in_bytes:
1185 00008A83 B9[2B420100]      <1>       mov    ecx, VolSize_Bytes
1186 00008A88 EB0D              <1>       jmp    short loc_write_vol_fspace_str
1187                            <1> loc_vol_fspace_in_kbytes:
1188 00008A8A 66BB0004          <1>       mov    bx, 1024
1189 00008A8E F7F3              <1>       div    ebx
1190 00008A90 B9[1E420100]      <1>       mov    ecx, VolSize_KiloBytes
1191 00008A95 31D2              <1>       xor    edx, edx ; 0
1192                            <1> loc_write_vol_fspace_str:
1193 00008A97 890D[17920100]    <1>       mov    [VolSize_Unit2], ecx
1194                            <1>       ;
1195 00008A9D BF[39920100]      <1>       mov    edi, Vol_Free_Sectors_Str_End
1196                            <1>         ;mov byte [edi], 0
1197 00008AA2 B90A000000        <1>       mov    ecx, 10
1198                            <1> loc_write_vol_fspace_chr:
1199 00008AA7 F7F1              <1>       div    ecx
1200 00008AA9 80C230            <1>       add    dl, '0'
1201 00008AAC 4F                <1>       dec    edi
1202 00008AAD 8817              <1>       mov    [edi], dl
1203 00008AAF 85C0              <1>       test   eax, eax
```

```
1204 00008AB1 7404                <1>       jz     short loc_write_vol_fspace_str_ok
1205 00008AB3 28D2                <1>       sub    dl, dl ; 0
1206 00008AB5 EBF0                <1>       jmp    short loc_write_vol_fspace_chr
1207                              <1>
1208                              <1> loc_write_vol_fspace_str_ok:
1209 00008AB7 893D[2B920100]      <1>       mov    [Vol_Free_Sectors_Str_Start], edi
1210                              <1>       ;
1211 00008ABD BE[34420100]        <1>       mov    esi, Volume_in_drive
1212 00008AC2 E89EEAFFFF          <1>       call   print_msg
1213 00008AC7 BE[74420100]        <1>       mov    esi, Vol_Name
1214 00008ACC E894EAFFFF          <1>       call   print_msg
1215 00008AD1 BE[434D0100]        <1>       mov    esi, nextline
1216 00008AD6 E88AEAFFFF          <1>       call   print_msg
1217                              <1>       ;
1218 00008ADB BE[D5420100]        <1>       mov    esi, Vol_Total_Sector_Header
1219 00008AE0 E880EAFFFF          <1>       call   print_msg
1220 00008AE5 8B35[1B920100]      <1>       mov    esi, [Vol_Tot_Sec_Str_Start]
1221 00008AEB E875EAFFFF          <1>       call   print_msg
1222 00008AF0 8B35[13920100]      <1>       mov    esi, [VolSize_Unit1]
1223 00008AF6 E86AEAFFFF          <1>       call   print_msg
1224                              <1>       ;
1225 00008AFB BE[E6420100]        <1>       mov    esi, Vol_Free_Sectors_Header
1226 00008B00 E860EAFFFF          <1>       call   print_msg
1227 00008B05 8B35[2B920100]      <1>       mov    esi, [Vol_Free_Sectors_Str_Start]
1228 00008B0B E855EAFFFF          <1>       call   print_msg
1229 00008B10 8B35[17920100]      <1>       mov    esi, [VolSize_Unit2]
1230 00008B16 E84AEAFFFF          <1>       call   print_msg
1231                              <1>       ;
1232 00008B1B 5E                  <1>       pop    esi
1233                              <1>
1234                              <1>       ;mov   ah, [esi+LD_FSType]
1235                              <1>       ;mov   al, [esi+LD_FATType]
1236 00008B1C 668B4603            <1>       mov    ax, [esi+LD_FATType]
1237                              <1>
1238 00008B20 C3                  <1>       retn
1239                              <1>
1240                              <1> move_volume_name_and_serial_no:
1241                              <1>       ; 08/02/2016  (TRDOS 386 = TRDOS v2.0)
1242                              <1>       ; this routine will be called by
1243                              <1>       ; "print_volume_info" and "print_directory"
1244                              <1>       ; INPUT ->
1245                              <1>       ;     ESI = Logical DOS drv descripton table address
1246                              <1>       ; OUTPUT ->
1247                              <1>       ;     *Volume name will be moved to text area
1248                              <1>       ;     *Volume serial number will be converted to
1249                              <1>       ;      text and will be moved to text area
1250                              <1>       ;   cf = 1 -> invalid/unknown dos drive
1251                              <1>       ;   cf = 0 -> ecx = 0
1252                              <1>       ;
1253                              <1>       ; (eax, edx, ecx, esi, edi will be changed)
1254                              <1>
1255 00008B21 BF[74420100]        <1>       mov    edi, Vol_Name
1256                              <1>
1257                              <1>       ;mov   ah, [esi+LD_FSType]
1258                              <1>       ;mov   al, [esi+LD_FATType]
1259 00008B26 668B4603            <1>       mov    ax, [esi+LD_FATType]
1260 00008B2A 80FCA1              <1>       cmp    ah, 0A1h
1261 00008B2D 7418                <1>       je     short mvn_2
1262 00008B2F 08E4                <1>       or     ah, ah
1263 00008B31 7404                <1>       jz     short mvn_0
1264 00008B33 08C0                <1>       or     al, al
1265 00008B35 7504                <1>       jnz    short mvn_1
1266                              <1> mvn_0:
1267 00008B37 8A06                <1>       mov    al, [esi]
1268 00008B39 F9                  <1>       stc
1269 00008B3A C3                  <1>       retn
1270                              <1> mvn_1:
1271 00008B3B 3C02                <1>       cmp    al, 2
1272 00008B3D 7717                <1>       ja     short mvn_3
1273                              <1>       ;or    al, al
1274                              <1>       ;jz    short mvn_2
1275 00008B3F 8B462D              <1>       mov    eax, [esi+LD_BPB+VolumeID]
1276 00008B42 83C631              <1>       add    esi, LD_BPB+VolumeLabel
1277 00008B45 EB15                <1>       jmp    short mvn_4
1278                              <1> mvn_2:
1279 00008B47 8B4628              <1>       mov    eax, [esi+LD_FS_VolumeSerial]
1280 00008B4A 83C62C              <1>       add    esi, LD_FS_VolumeName
1281 00008B4D B910000000          <1>       mov    ecx, 16
1282 00008B52 F3A5                <1>       rep    movsd
1283 00008B54 EB10                <1>       jmp    short mvn_5
1284                              <1> mvn_3:
1285 00008B56 8B4649              <1>       mov    eax, [esi+LD_BPB+FAT32_VolID]
1286 00008B59 83C64D              <1>       add    esi, LD_BPB+FAT32_VolLab
1287                              <1> mvn_4:
1288 00008B5C B90B000000          <1>       mov    ecx, 11
1289 00008B61 F3A4                <1>       rep    movsb
1290 00008B63 C60700              <1>       mov    byte [edi], 0
1291                              <1> mvn_5:
1292                              <1>       ;mov   [Current_VolSerial], eax
1293 00008B66 E8AAB7FFFF          <1>       call   dwordtohex
1294 00008B6B 8915[C9420100]      <1>       mov    [Vol_Serial1], edx
1295 00008B71 A3[CE420100]        <1>       mov    [Vol_Serial2], eax
1296                              <1>       ; ecx = 0
1297 00008B76 C3                  <1>       retn
1298                              <1>
1299                              <1> get_volume_serial_number:
1300                              <1>       ; 19/01/2016 (TRDOS 386 = TRDOS v2.0)
1301                              <1>       ; 08/08/2010
1302                              <1>       ;
1303                              <1>       ; INPUT -> DL = Logical DOS Drive number
1304                              <1>       ; OUTPUT -> EAX = Volume serial number
1305                              <1>       ;          BL= FAT Type
1306                              <1>       ;          BH = Logical DOS drv Number (DL input)
1307                              <1>       ; cf = 1 -> Drive not ready
1308                              <1>
```

```
1309 00008B77 31DB                <1>       xor    ebx, ebx
1310 00008B79 88D7                <1>       mov    bh, dl
1311 00008B7B 3815[A5400100]      <1>       cmp    [Last_DOS_DiskNo], dl
1312 00008B81 7304                <1>       jnb    short loc_gvsn_start
1313                              <1> loc_gvsn_stc_retn:
1314 00008B83 31C0                <1>       xor    eax, eax
1315 00008B85 F9                  <1>       stc
1316 00008B86 C3                  <1>         retn
1317                              <1> loc_gvsn_start:
1318 00008B87 56                  <1>       push   esi
1319 00008B88 BE00010900          <1>       mov    esi, Logical_DOSDisks
1320 00008B8D 01DE                <1>       add    esi, ebx
1321 00008B8F 8A5E03              <1>       mov    bl, [esi+LD_FATType]
1322 00008B92 20DB                <1>       and    bl, bl
1323 00008B94 740F                <1>       jz     short loc_gvsn_fs
1324 00008B96 80FB02              <1>       cmp    bl, 2
1325 00008B99 7705                <1>       ja     short loc_gvsn_fat32
1326                              <1> loc_gvsn_fat:
1327 00008B9B 83C62D              <1>       add    esi, LD_BPB + VolumeID
1328 00008B9E EB0E                <1>       jmp    short loc_gvsn_return
1329                              <1> loc_gvsn_fat32:
1330 00008BA0 83C649              <1>       add    esi, LD_BPB + FAT32_VolID
1331 00008BA3 EB09                <1>       jmp    short loc_gvsn_return
1332                              <1> loc_gvsn_fs:
1333 00008BA5 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
1334 00008BA9 75D8                <1>       jne    short loc_gvsn_stc_retn
1335 00008BAB 83C628              <1>       add    esi, LD_FS_VolumeSerial
1336                              <1> loc_gvsn_return:
1337 00008BAE 8B06                <1>       mov    eax, [esi]
1338 00008BB0 5E                  <1>       pop    esi
1339 00008BB1 C3                  <1>       retn
1340                              <1>
1341                              <1> ; CMD_INTR.ASM [ TRDOS Command Interpreter Procedure ]
1342                              <1> ; 09/11/2011
1343                              <1> ; 29/01/2005
1344                              <1>
1345                              <1> command_interpreter:
1346                              <1>       ; 16/10/2016
1347                              <1>       ; 12/10/2016
1348                              <1>       ; 13/05/2016
1349                              <1>       ; 07/05/2016
1350                              <1>       ; 04/03/2016
1351                              <1>       ; 04/02/2016
1352                              <1>       ; 03/02/2016
1353                              <1>       ; 30/01/2016
1354                              <1>       ; 29/01/2016 (TRDOS 386 = TRDOS 2.0)
1355                              <1>       ; 15/09/2011
1356                              <1>       ; 29/01/2005
1357                              <1>
1358                              <1>       ; Input: ecx = command word length (CL)
1359                              <1>       ;       CommandBuffer = Command string offset
1360                              <1>
1361 00008BB2 C605[CC920100]00    <1>       mov    byte [Program_Exit],0
1362 00008BB9 80F904              <1>       cmp    cl, 4
1363 00008BBC 0F87B5020000        <1>         ja      c_6
1364 00008BC2 0F8237010000        <1>         jb      c_2
1365                              <1> c_4:
1366                              <1>
1367                              <1> cmp_cmd_exit:
1368 00008BC8 BF[13410100]        <1>       mov    edi, Cmd_Exit
1369 00008BCD E8C2030000          <1>       call   cmp_cmd
1370 00008BD2 7208                <1>       jc     short cmp_cmd_date
1371                              <1>
1372 00008BD4 C605[CC920100]01    <1>         mov     byte [Program_Exit], 1
1373 00008BDB C3                  <1>         retn
1374                              <1>
1375                              <1> cmp_cmd_date:
1376 00008BDC B104                <1>       mov    cl, 4
1377 00008BDE BF[2F410100]        <1>       mov    edi, Cmd_Date
1378 00008BE3 E8AC030000          <1>       call   cmp_cmd
1379 00008BE8 720B                <1>         jc   short cmp_cmd_time
1380                              <1>
1381 00008BEA E8D0F7FFFF          <1>       call   show_date
1382 00008BEF E80FF8FFFF          <1>       call   set_date
1383 00008BF4 C3                  <1>       retn
1384                              <1>
1385                              <1> cmp_cmd_time:
1386 00008BF5 B104                <1>       mov    cl, 4
1387 00008BF7 BF[34410100]        <1>       mov    edi, Cmd_Time
1388 00008BFC E893030000          <1>       call   cmp_cmd
1389 00008C01 720B                <1>       jc     short cmp_cmd_show
1390                              <1>
1391 00008C03 E8C6FAFFFF          <1>       call   show_time
1392 00008C08 E8F8FAFFFF          <1>       call   set_time
1393 00008C0D C3                  <1>       retn
1394                              <1>
1395                              <1> cmp_cmd_show:
1396 00008C0E B104                <1>       mov    cl, 4
1397 00008C10 BF[45410100]        <1>       mov    edi, Cmd_Show
1398 00008C15 E87A030000          <1>       call   cmp_cmd
1399 00008C1A 0F83050A0000        <1>         jnc     show_file
1400                              <1>
1401                              <1> cmp_cmd_echo:
1402 00008C20 B104                <1>       mov    cl, 4
1403 00008C22 BF[81410100]        <1>       mov    edi, Cmd_Echo
1404 00008C27 E868030000          <1>       call   cmp_cmd
1405 00008C2C 7224                <1>       jc     short cmp_cmd_copy
1406                              <1>
1407                              <1>       ; 22/11/2017
1408                              <1>       ; AL = 0
1409 00008C2E 803E20              <1>       cmp    byte [esi], 20h
1410 00008C31 7215                <1>       jb     short cmd_echo_nextline
1411                              <1>       ; 14/04/2016
1412 00008C33 56                  <1>       push   esi
1413                              <1> cmd_echo_asciiz:
```

```
                        <1>       ;inc   esi
                        <1>       ;mov   al, [esi]
                        <1>       ; 22/11/2017
00008C34 AC             <1>       lodsb
00008C35 3C20           <1>       cmp    al, 20h
00008C37 73FB           <1>       jnb    short cmd_echo_asciiz
00008C39 4E             <1>       dec    esi
00008C3A C60600         <1>       mov    byte [esi], 0
00008C3D 5E             <1>       pop    esi
00008C3E 89F7           <1>       mov    edi, esi
00008C40 E820E9FFFF     <1>       call   print_msg
00008C45 C60700         <1>       mov    byte [edi], 0
                        <1> cmd_echo_nextline:
00008C48 BE[B14D0100]   <1>       mov    esi, NextLine
                        <1>       ;call  print_msg
                        <1>       ;retn
00008C4D E913E9FFFF     <1>       jmp    print_msg
                        <1>
                        <1> cmp_cmd_copy:
00008C52 B104           <1>       mov    cl, 4
00008C54 BF[68410100]   <1>       mov    edi, Cmd_Copy
00008C59 E836030000     <1>       call   cmp_cmd
00008C5E 0F83CC170000   <1>       jnc    copy_file
                        <1>
                        <1> cmp_cmd_move:
00008C64 B104           <1>       mov    cl, 4
00008C66 BF[6D410100]   <1>       mov    edi, Cmd_Move
00008C6B E824030000     <1>       call   cmp_cmd
00008C70 0F836E160000   <1>       jnc    move_file
                        <1>
                        <1> cmp_cmd_path:
00008C76 B104           <1>       mov    cl, 4
00008C78 BF[72410100]   <1>       mov    edi, Cmd_Path
00008C7D E812030000     <1>       call   cmp_cmd
00008C82 0F83F0190000   <1>       jnc    set_get_path
                        <1>
                        <1> cmp_cmd_beep:
00008C88 B104           <1>       mov    cl, 4
00008C8A BF[9F410100]   <1>       mov    edi, Cmd_Beep
00008C8F E800030000     <1>       call   cmp_cmd
00008C94 720B           <1>       jc     short cmp_cmd_find
                        <1>       ; 13/05/2016
00008C96 8A3D[468A0100] <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
00008C9C E95097FFFF     <1>       jmp    beeper
                        <1>
                        <1> cmp_cmd_find:
00008CA1 B104           <1>       mov    cl, 4
00008CA3 BF[7C410100]   <1>       mov    edi, Cmd_Find
00008CA8 E8E7020000     <1>       call   cmp_cmd
00008CAD 0F82C4020000   <1>         jc     cmp_cmd_external
                        <1>
                        <1>       ;call  find_and_list_files
00008CB3 E9AF220000     <1>       jmp    find_and_list_files
                        <1>       ;retn
                        <1>
                        <1> c_1:
00008CB8 AD             <1>       lodsd
                        <1> cmp_cmd_help:
00008CB9 3C3F           <1>       cmp    al, '?'
00008CBB 751D           <1>         jne    short cmp_cmd_remark
                        <1>
00008CBD BE[05410100]   <1>       mov    esi, Command_List
                        <1> cmd_help_next_w:
00008CC2 E89EE8FFFF     <1>       call   print_msg
                        <1>
00008CC7 803E20         <1>       cmp    byte [esi], 20h ; 0
00008CCA 7232           <1>       jb     short cmd_help_retn
                        <1>
00008CCC 56             <1>       push   esi
00008CCD BE[434D0100]   <1>       mov    esi, nextline
00008CD2 E88EE8FFFF     <1>       call   print_msg
00008CD7 5E             <1>       pop    esi
00008CD8 EBE8           <1>       jmp    short cmd_help_next_w
                        <1>
                        <1> cmp_cmd_remark:
00008CDA 3C2A           <1>       cmp    al, '*'
00008CDC 0F8595020000   <1>         jne    cmp_cmd_external
00008CE2 46             <1>       inc    esi
00008CE3 BF[408B0100]   <1>       mov    edi, Remark
00008CE8 8A06           <1>       mov    al, [esi]
00008CEA 3C20           <1>       cmp    al, 20h
00008CEC 7707           <1>       ja     short cmd_remark_write
00008CEE 89FE           <1>       mov    esi, edi ; Remark
00008CF0 E970E8FFFF     <1>       jmp    print_msg
                        <1>
                        <1> cmd_remark_write:
00008CF5 AA             <1>       stosb
00008CF6 AC             <1>       lodsb
00008CF7 3C20           <1>       cmp    al, 20h
00008CF9 73FA           <1>       jnb    short cmd_remark_write
00008CFB C60700         <1>       mov    byte [edi], 0
                        <1>
                        <1> cmd_help_retn:
                        <1> cmd_remark_retn:
                        <1> cd_retn:
00008CFE C3             <1>       retn
                        <1>
                        <1> c_2:
00008CFF 80F902         <1>       cmp    cl, 2
00008D02 0F87AF000000   <1>         ja     c_3
00008D08 BE[8E8B0100]   <1>       mov    esi, CommandBuffer
00008D0D 72A9           <1>       jb     short c_1
                        <1>
                        <1> cmp_cmd_cd:
00008D0F 66AD           <1>       lodsw
```

```
1519 00008D11 663D4344            <1>       cmp    ax, 'CD'
1520 00008D15 7551                <1>       jne    short cmp_cmd_drive
1521 00008D17 46                  <1>         inc  esi
1522                              <1> cd_0:
1523 00008D18 668B06              <1>       mov    ax, [esi]
1524 00008D1B 3C20                <1>       cmp    al, 20h
1525 00008D1D 76DF                <1>       jna    short cd_retn
1526                              <1>       ; 10/02/2016
1527 00008D1F 80FC3A              <1>       cmp    ah, ':'
1528 00008D22 7504                <1>       jne    short cd_1
1529 00008D24 46                  <1>       inc    esi
1530 00008D25 46                  <1>       inc    esi
1531 00008D26 EB49                <1>       jmp    short cd_2
1532                              <1>
1533                              <1> cd_1: ; change current directory
1534                              <1>       ; 29/11/2009
1535                              <1>       ; AH = CDh   ; to separate 'CD' command from others
1536                              <1>                    ; for restoring current directory
1537                              <1>                    ; 0CDh sign is for saving cdir into
1538                              <1>                    ; DOS drv description table cdir area
1539                              <1>
1540 00008D28 B4CD                <1>       mov    ah, 0CDh ; mov byte [CD_COMMAND], 0CDh
1541                              <1>
1542 00008D2A E81D230000          <1>       call   change_current_directory
1543 00008D2F 0F8337220000        <1>         jnc     change_prompt_dir_string
1544                              <1>
1545                              <1> cd_error_messages:
1546 00008D35 3C03                <1>       cmp    al, 3
1547 00008D37 740C                <1>       je     short cd_path_not_found
1548                              <1>       ; 16/10/2016 (15h -> 15)
1549 00008D39 3C0F                <1>       cmp    al, 15 ; drive not ready error
1550 00008D3B 7459                <1>       je     short cd_drive_not_ready
1551 00008D3D 3C11                <1>       cmp    al, 17 ; read error
1552 00008D3F 7455                <1>       je     short cd_drive_not_ready
1553 00008D41 3C13                <1>       cmp    al, 19 ; ; Bad directory/path name
1554 00008D43 7466                <1>       je     short cd_command_failed
1555                              <1>
1556                              <1> cd_path_not_found:
1557 00008D45 50                  <1>       push   eax ; 29/12/2017
1558                              <1>       ;push  ax
1559 00008D46 BE[A8430100]        <1>       mov    esi, Msg_Dir_Not_Found
1560 00008D4B E815E8FFFF          <1>       call   print_msg
1561                              <1>       ;pop   ax
1562 00008D50 58                  <1>       pop    eax ; 29/12/2017
1563 00008D51 3A25[DC8A0100]      <1>       cmp    ah, [Current_Dir_Level]
1564 00008D57 0F830F220000        <1>         jnb     change_prompt_dir_string
1565 00008D5D 8825[DC8A0100]      <1>       mov    [Current_Dir_Level], ah
1566 00008D63 E904220000          <1>         jmp     change_prompt_dir_string
1567                              <1>
1568                              <1> cmp_cmd_drive: ; change current drive
1569                              <1>       ; C:, D:, E: etc.
1570 00008D68 80FC3A              <1>       cmp    ah, ':'
1571 00008D6B 0F8506020000        <1>         jne     cmp_cmd_external
1572                              <1>
1573                              <1> cd_2: ; 'CD C:', 'CD D:' ...
1574 00008D71 803E20              <1>       cmp    byte [esi], 20h
1575 00008D74 0F8707020000        <1>         ja      loc_cmd_failed
1576                              <1>
1577 00008D7A 24DF                <1>       and    al, 0DFh
1578 00008D7C 2C41                <1>       sub    al, 'A'
1579 00008D7E 0F82FD010000        <1>         jc      loc_cmd_failed
1580                              <1>
1581 00008D84 3A05[A5400100]      <1>         cmp     al, [Last_DOS_DiskNo]
1582 00008D8A 770A                <1>         ja   short cd_drive_not_ready
1583                              <1>
1584 00008D8C 88C2                <1>       mov    dl, al
1585 00008D8E E85BF3FFFF          <1>       call   change_current_drive
1586 00008D93 7201                <1>       jc     short cd_drive_not_ready
1587 00008D95 C3                  <1>       retn
1588                              <1>
1589                              <1> cd_drive_not_ready:
1590 00008D96 BE[65430100]        <1>       mov    esi, Msg_Not_Ready_Read_Err
1591 00008D9B E8C5E7FFFF          <1>       call   print_msg
1592                              <1>
1593                              <1> cd_fail_drive_restart:
1594 00008DA0 8A15[DE8A0100]      <1>       mov    dl, [Current_Drv]
1595                              <1>       ;call  change_current_drive
1596 00008DA6 E943F3FFFF          <1>         jmp     change_current_drive
1597                              <1>       ;retn
1598                              <1>
1599                              <1> cd_command_failed:
1600 00008DAB BE[46430100]        <1>       mov    esi, Msg_Bad_Command
1601 00008DB0 E8B0E7FFFF          <1>       call   print_msg
1602 00008DB5 EBE9                <1>       jmp    short cd_fail_drive_restart
1603                              <1>
1604                              <1> c_3:
1605                              <1> cmp_cmd_dir:
1606 00008DB7 BF[05410100]        <1>       mov    edi, Cmd_Dir
1607 00008DBC E8D3010000          <1>       call   cmp_cmd
1608 00008DC1 0F8380020000        <1>       jnc    print_directory_list
1609                              <1>
1610                              <1> cmp_cmd_cls:
1611 00008DC7 B103                <1>       mov    cl, 3
1612 00008DC9 BF[41410100]        <1>       mov    edi, Cmd_Cls
1613 00008DCE E8C1010000          <1>       call   cmp_cmd
1614 00008DD3 0F83A2E7FFFF        <1>         jnc  clear_screen
1615                              <1>
1616                              <1> cmp_cmd_ver:
1617 00008DD9 B103                <1>       mov    cl, 3
1618 00008DDB BF[0F410100]        <1>       mov    edi, Cmd_Ver
1619 00008DE0 E8AF010000          <1>       call   cmp_cmd
1620 00008DE5 720A                <1>       jc     short cmp_cmd_mem
1621                              <1>
1622 00008DE7 BE[AD400100]        <1>       mov    esi, mainprog_Version
1623                              <1>       ;call  print_msg
```

```
1624 00008DEC E974E7FFFF          <1>       jmp    print_msg
1625                              <1>       ;retn
1626                              <1>
1627                              <1> cmp_cmd_mem:
1628 00008DF1 B103                <1>       mov    cl, 3
1629 00008DF3 BF[77410100]        <1>       mov    edi, Cmd_Mem
1630 00008DF8 E897010000          <1>       call   cmp_cmd
1631 00008DFD 0F8344B4FFFF        <1>       jnc    memory_info
1632                              <1>
1633                              <1> cmp_cmd_del:
1634 00008E03 B103                <1>       mov    cl, 3
1635 00008E05 BF[4A410100]        <1>       mov    edi, Cmd_Del
1636 00008E0A E885010000          <1>       call   cmp_cmd
1637 00008E0F 0F83280F0000        <1>         jnc     delete_file
1638                              <1>
1639                              <1> cmp_cmd_set:
1640 00008E15 B103                <1>       mov    cl, 3
1641 00008E17 BF[3D410100]        <1>       mov    edi, Cmd_Set
1642 00008E1C E873010000          <1>       call   cmp_cmd
1643 00008E21 0F83C9170000        <1>         jnc     set_get_env
1644                              <1>
1645                              <1> cmp_cmd_run:
1646 00008E27 B103                <1>       mov    cl, 3
1647 00008E29 BF[39410100]        <1>       mov    edi, Cmd_Run
1648 00008E2E E861010000          <1>       call   cmp_cmd
1649                              <1>       ; 07/05/2016
1650 00008E33 0F823E010000        <1>         jc      cmp_cmd_external
1651 00008E39 E90F1E0000          <1>       jmp    load_and_execute_file
1652                              <1> c_5:
1653                              <1> cmp_cmd_mkdir:
1654 00008E3E BF[62410100]        <1>       mov    edi, Cmd_Mkdir
1655 00008E43 E84C010000          <1>       call   cmp_cmd
1656 00008E48 0F83990A0000        <1>         jnc     make_directory
1657                              <1>
1658                              <1> cmp_cmd_rmdir:
1659 00008E4E B105                <1>       mov    cl, 5
1660 00008E50 BF[5C410100]        <1>       mov    edi, Cmd_Rmdir
1661 00008E55 E83A010000          <1>       call   cmp_cmd
1662 00008E5A 0F83AA0B0000        <1>         jnc     delete_directory
1663                              <1>
1664                              <1> cmp_cmd_chdir:
1665 00008E60 B105                <1>       mov    cl, 5
1666 00008E62 BF[99410100]        <1>       mov    edi, Cmd_Chdir
1667 00008E67 E828010000          <1>       call   cmp_cmd
1668 00008E6C 0F8205010000        <1>         jc      cmp_cmd_external
1669                              <1>
1670 00008E72 E9A1FEFFFF          <1>       jmp    cd_0
1671                              <1>
1672                              <1> c_6:
1673 00008E77 80F906              <1>       cmp    cl, 6
1674 00008E7A 0F87E0000000        <1>         ja      c_8
1675 00008E80 72BC                <1>       jb     short c_5
1676                              <1> cmp_cmd_prompt:
1677 00008E82 BF[18410100]        <1>       mov    edi, Cmd_Prompt
1678 00008E87 E808010000          <1>       call   cmp_cmd
1679 00008E8C 722F                <1>         jc   short cmp_cmd_volume
1680                              <1> get_prompt_name_fchar:
1681 00008E8E AC                  <1>       lodsb
1682 00008E8F 3C20                <1>       cmp    al, 20h
1683 00008E91 74FB                <1>       je     short get_prompt_name_fchar
1684 00008E93 7713                <1>       ja     short loc_change_prompt_label
1685                              <1> default_command_prompt: ; 31/12/2017 ('sysprompt')
1686 00008E95 BE[F9400100]        <1>       mov    esi, TRDOSPromptLabel
1687 00008E9A C7065452444F        <1>       mov    dword [esi], "TRDO"
1688 00008EA0 66C746045300        <1>              mov    word [esi+4], "S"
1689                              <1> loc_cmd_prompt_return:
1690 00008EA6 C3                  <1>       retn
1691                              <1>
1692                              <1> set_command_prompt: ; 31/12/2017 ('sysprompt')
1693 00008EA7 AC                  <1>       lodsb
1694                              <1> loc_change_prompt_label:
1695 00008EA8 66B90B00            <1>       mov    cx, 11
1696 00008EAC BF[F9400100]        <1>       mov    edi, TRDOSPromptLabel
1697                              <1> put_char_new_prompt_label:
1698 00008EB1 AA                  <1>       stosb
1699 00008EB2 AC                  <1>       lodsb
1700 00008EB3 3C20                <1>       cmp    al, 20h
1701 00008EB5 7202                <1>       jb     short pass_put_new_prompt_label
1702 00008EB7 E2F8                <1>       loop   put_char_new_prompt_label
1703                              <1> pass_put_new_prompt_label:
1704 00008EB9 C60700              <1>       mov    byte [edi], 0
1705 00008EBC C3                  <1>       retn
1706                              <1>
1707                              <1> cmp_cmd_volume:
1708 00008EBD B106                <1>       mov    cl, 6
1709 00008EBF BF[1F410100]        <1>       mov    edi, Cmd_Volume
1710 00008EC4 E8CB000000          <1>       call   cmp_cmd
1711 00008EC9 7255                <1>         jc   short cmp_cmd_attrib
1712                              <1>
1713                              <1> cmd_vol1:
1714 00008ECB AC                  <1>       lodsb
1715 00008ECC 3C20                <1>       cmp    al, 20h
1716 00008ECE 7707                <1>       ja     short cmd_vol2
1717 00008ED0 A0[DE8A0100]        <1>       mov    al, [Current_Drv]
1718 00008ED5 EB3D                <1>       jmp    short cmd_vol4
1719                              <1> cmd_vol2:
1720 00008ED7 3C41                <1>       cmp    al, 'A'
1721 00008ED9 0F82A2000000        <1>         jb      loc_cmd_failed
1722 00008EDF 3C7A                <1>       cmp    al, 'z'
1723 00008EE1 0F879A000000        <1>         ja      loc_cmd_failed
1724 00008EE7 3C5A                <1>       cmp    al, 'Z'
1725 00008EE9 760A                <1>       jna    short cmd_vol3
1726 00008EEB 3C61                <1>       cmp    al, 'a'
1727 00008EED 0F828E000000        <1>         jb      loc_cmd_failed
1728 00008EF3 24DF                <1>       and    al, 0DFh
```

```
1729                                    <1> cmd_vol3:
1730 00008EF5 8A26                     <1>         mov     ah, [esi]
1731 00008EF7 80FC3A                   <1>         cmp     ah, ':'
1732 00008EFA 0F8581000000             <1>           jne     loc_cmd_failed
1733 00008F00 2C41                     <1>         sub     al, 'A'
1734 00008F02 3A05[A5400100]           <1>           cmp     al, [Last_DOS_DiskNo]
1735 00008F08 760A                     <1>         jna     short cmd_vol4
1736                                    <1>
1737 00008F0A BE[65430100]             <1>         mov     esi, Msg_Not_Ready_Read_Err
1738 00008F0F E951E6FFFF               <1>         jmp     print_msg
1739                                    <1>
1740                                    <1> cmd_vol4:
1741 00008F14 E88EFAFFFF               <1>         call    print_volume_info
1742 00008F19 0F8277FEFFFF             <1>           jc      cd_drive_not_ready
1743 00008F1F C3                       <1>         retn
1744                                    <1>
1745                                    <1> cmp_cmd_attrib:
1746 00008F20 B106                     <1>         mov     cl, 6
1747 00008F22 BF[4E410100]             <1>         mov     edi, Cmd_Attrib
1748 00008F27 E868000000               <1>         call    cmp_cmd
1749 00008F2C 0F831D0F0000             <1>           jnc     set_file_attributes
1750                                    <1>
1751                                    <1> cmp_cmd_rename:
1752 00008F32 B106                     <1>         mov     cl, 6
1753 00008F34 BF[55410100]             <1>         mov     edi, Cmd_Rename
1754 00008F39 E856000000               <1>         call    cmp_cmd
1755 00008F3E 0F8353110000             <1>           jnc     rename_file
1756                                    <1>
1757                                    <1> cmp_cmd_device:
1758 00008F44 B106                     <1>         mov     cl, 6
1759 00008F46 BF[8A410100]             <1>         mov     edi, Cmd_Device
1760 00008F4B E844000000               <1>         call    cmp_cmd
1761 00008F50 7225                     <1>           jc    short cmp_cmd_external
1762                                    <1>
1763 00008F52 C3                       <1>         retn
1764                                    <1>
1765                                    <1> c_7:
1766                                    <1> cmp_cmd_devlist:
1767 00008F53 BF[91410100]             <1>         mov     edi, Cmd_DevList
1768 00008F58 E837000000               <1>         call    cmp_cmd
1769 00008F5D 7218                     <1>           jc    short cmp_cmd_external
1770                                    <1>
1771                                    <1> loc_cmd_return:
1772 00008F5F C3                       <1>         retn
1773                                    <1>
1774                                    <1> c_8:
1775 00008F60 80F908                   <1>           cmp  cl, 8
1776 00008F63 7712                     <1>         ja      short cmp_cmd_external
1777 00008F65 72EC                     <1>         jb      short c_7
1778                                    <1>
1779                                    <1> cmp_cmd_longname:
1780 00008F67 BF[26410100]             <1>         mov     edi, Cmd_LongName
1781 00008F6C E823000000               <1>         call    cmp_cmd
1782 00008F71 0F8350060000             <1>           jnc     get_and_print_longname
1783                                    <1>
1784                                    <1> cmp_cmd_external:
1785                                    <1>         ; 07/05/2016
1786                                    <1>         ; 22/04/2016
1787 00008F77 BE[8E8B0100]             <1>         mov     esi, CommandBuffer
1788 00008F7C E9CC1C0000               <1>         jmp     loc_run_check_filename
1789                                    <1>
1790                                    <1> loc_cmd_failed:
1791 00008F81 803D[8E8B0100]20         <1>         cmp     byte [CommandBuffer], 20h
1792 00008F88 76D5                     <1>         jna     short loc_cmd_return
1793 00008F8A BE[46430100]             <1>         mov     esi, Msg_Bad_Command
1794                                    <1> ;       call    print_msg
1795                                    <1> ;loc_cmd_return:
1796                                    <1> ;       retn
1797 00008F8F E9D1E5FFFF               <1>         jmp     print_msg
1798                                    <1>
1799                                    <1> cmp_cmd:
1800                                    <1>           ; 29/01/2016 (TRDOS 386 = TRDOS v2.0)
1801 00008F94 BE[8E8B0100]             <1>           mov esi, CommandBuffer
1802                                    <1>           ; edi = internal command word (ASCIIZ)
1803                                    <1>         ; ecx = command length (<=8)
1804                                    <1> cmp_cmd_1:
1805 00008F99 AC                       <1>         lodsb
1806 00008F9A AE                       <1>         scasb
1807 00008F9B 750D                     <1>         jne     short cmp_cmd_3
1808 00008F9D E2FA                     <1>         loop    cmp_cmd_1
1809 00008F9F AC                       <1>         lodsb
1810 00008FA0 3C20                     <1>         cmp     al, 20h
1811 00008FA2 7703                     <1>         ja      short cmp_cmd_2
1812 00008FA4 30C0                     <1>         xor     al, al
1813                                    <1>         ; ZF = 1 -> internal command word matches
1814 00008FA6 C3                       <1>         retn
1815                                    <1> cmp_cmd_2:
1816                                    <1>         ; ZF = 0 (CF = 0) -> external command word
1817 00008FA7 58                       <1>         pop     eax ; no return to the caller from here
1818 00008FA8 EBCD                     <1>         jmp     cmp_cmd_external
1819                                    <1> cmp_cmd_3:
1820 00008FAA F9                       <1>         stc
1821                                    <1>         ; CF = 1 -> internal command word does not match
1822 00008FAB C3                       <1>         retn
1823                                    <1>
1824                                    <1> loc_run_cmd_failed:
1825                                    <1>         ; 15/03/2016
1826                                    <1>         ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
1827                                    <1>         ; 07/12/2009 (CMD_INTR.ASM)
1828                                    <1>         ; 29/11/2009
1829                                    <1>
1830 00008FAC E863000000               <1>         call    restore_cdir_after_cmd_fail
1831                                    <1>
1832                                    <1> loc_run_cmd_failed_cmp_al:
1833                                    <1>         ; End of Restore_CDIR code (29/11/2009)
```

```
1834                                    <1>
1835 00008FB1 3C01                     <1>       cmp    al, 1 ; Bad command or file name
1836 00008FB3 74CC                     <1>       je     loc_cmd_failed
1837                                    <1> loc_run_dir_not_found:
1838 00008FB5 3C03                     <1>       cmp    al, 3
1839 00008FB7 750A                     <1>       jne    short loc_run_file_notfound_msg
1840                                    <1>       ; Path not found (MS-DOS Error Code = 3)
1841 00008FB9 BE[A8430100]             <1>       mov    esi, Msg_Dir_Not_Found
1842 00008FBE E9A2E5FFFF               <1>       jmp    print_msg
1843                                    <1>
1844                                    <1> loc_run_file_notfound_msg:
1845 00008FC3 3C02                     <1>       cmp    al, 2 ; File not found
1846 00008FC5 750A                     <1>       jne    short loc_run_file_drv_read_err
1847                                    <1>
1848                                    <1> loc_print_file_notfound_msg:
1849 00008FC7 BE[BF430100]             <1>         mov     esi, Msg_File_Not_Found
1850                                    <1>       ;call  proc_printmsg
1851                                    <1>       ;retn
1852 00008FCC E994E5FFFF               <1>       jmp    print_msg
1853                                    <1>
1854                                    <1> loc_run_file_drv_read_err:
1855                                    <1>       ; Err: 17 (Read fault)
1856 00008FD1 3C11                     <1>       cmp    al, 17 ; Drive not ready or read error
1857 00008FD3 7404                     <1>       je     short loc_run_file_print_drv_read_err
1858                                    <1>       ;
1859 00008FD5 3C0F                     <1>       cmp    al, 15 ; Drive not ready (or read error)
1860 00008FD7 750A                     <1>       jne    short loc_run_file_toobig
1861                                    <1>
1862                                    <1> loc_run_file_print_drv_read_err:
1863 00008FD9 BE[65430100]             <1>       mov    esi, Msg_Not_Ready_Read_Err
1864 00008FDE E982E5FFFF               <1>       jmp    print_msg
1865                                    <1>
1866                                    <1> loc_run_file_toobig:
1867 00008FE3 3C08                     <1>       cmp    al, 8 ; Not enough free memory to load&run file
1868 00008FE5 750A                     <1>       jne    short loc_run_file_perm_denied
1869 00008FE7 BE[0A440100]             <1>       mov    esi, Msg_Insufficient_Memory
1870 00008FEC E974E5FFFF               <1>       jmp    print_msg
1871                                    <1>
1872                                    <1> loc_run_file_perm_denied:
1873                                    <1>       ; 29/12/2017
1874 00008FF1 3C0B                     <1>       cmp    al, ERR_PERM_DENIED ; 11 ; Permission denied
1875 00008FF3 750A                     <1>       jne    short loc_run_misc_error
1876 00008FF5 BE[9E450100]             <1>       mov    esi, Msg_Permission_Denied
1877 00008FFA E966E5FFFF               <1>       jmp    print_msg
1878                                    <1>
1879                                    <1>       ; 15/03/2016
1880                                    <1> print_misc_error_msg:
1881                                    <1> loc_run_misc_error:
1882                                    <1>       ; AL = Error code
1883 00008FFF E8D1B2FFFF               <1>       call   bytetohex
1884 00009004 66A3[3E440100]           <1>         mov     [error_code_hex], ax
1885                                    <1>
1886 0000900A BE[21440100]             <1>       mov    esi, Msg_Error_Code
1887                                    <1>       ;call  print_msg
1888                                    <1>       ;retn
1889                                    <1>
1890 0000900F E951E5FFFF               <1>       jmp    print_msg
1891                                    <1>
1892                                    <1> restore_cdir_after_cmd_fail:
1893                                    <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
1894 00009014 50                       <1>       push   eax
1895 00009015 8A3D[3A920100]           <1>       mov    bh, [RUN_CDRV] ; it is set at the beginning
1896                                    <1>                             ; of the 'run' command.
1897 0000901B 3A3D[DE8A0100]           <1>       cmp    bh, [Current_Drv]
1898 00009021 7409                     <1>       je     short loc_run_restore_cdir
1899 00009023 88FA                     <1>       mov    dl, bh
1900 00009025 E8C4F0FFFF               <1>       call   change_current_drive
1901 0000902A EB19                     <1>       jmp    short loc_run_err_pass_restore_cdir
1902                                    <1>
1903                                    <1> loc_run_restore_cdir:
1904 0000902C 803D[A6400100]00         <1>       cmp    byte [Restore_CDIR], 0
1905 00009033 7610                     <1>       jna    short loc_run_err_pass_restore_cdir
1906 00009035 30DB                     <1>       xor    bl, bl
1907 00009037 0FB7F3                   <1>       movzx  esi, bx
1908 0000903A 81C600010900             <1>       add    esi, Logical_DOSDisks
1909 00009040 E860F1FFFF               <1>       call   restore_current_directory
1910                                    <1>
1911                                    <1> loc_run_err_pass_restore_cdir:
1912 00009045 58                       <1>       pop    eax
1913 00009046 C3                       <1>       retn
1914                                    <1>
1915                                    <1> print_directory_list:
1916                                    <1>       ; 10/02/2016
1917                                    <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
1918                                    <1>       ; 06/12/2009 ('cmp_cmd_dir')
1919                                    <1>       ;
1920 00009047 66C705[7C930100]00-      <1>       mov    word [AttributesMask], 0800h ; ..except volume names..
1920 0000904F 08                       <1>
1921 00009050 A0[DE8A0100]             <1>       mov    al, [Current_Drv]
1922 00009055 A2[3A920100]             <1>       mov    [RUN_CDRV], al
1923                                    <1> get_dfname_fchar:
1924 0000905A AC                       <1>       lodsb
1925 0000905B 3C20                     <1>       cmp    al, 20h
1926 0000905D 74FB                     <1>       je     short get_dfname_fchar
1927 0000905F 0F82A4000000             <1>         jb      loc_print_dir_call_all
1928 00009065 3C2D                     <1>       cmp    al, '-'
1929 00009067 7542                     <1>       jne    short loc_print_dir_call_flt
1930                                    <1> get_next_attr_char:
1931 00009069 AC                       <1>       lodsb
1932 0000906A 3C20                     <1>       cmp    al, 20h
1933 0000906C 74FB                     <1>       je     short get_next_attr_char
1934 0000906E 0F820DFFFFFF             <1>         jb      loc_cmd_failed
1935 00009074 24DF                     <1>       and    al, 0DFh
1936 00009076 3C44                     <1>       cmp    al, 'D' ; directories only ?
1937 00009078 7512                     <1>       jne    short pass_only_directories
```

```
1938 0000907A AC                  <1>       lodsb
1939 0000907B 3C20                <1>       cmp    al, 20h
1940 0000907D 0F87FEFEFFFF        <1>         ja      loc_cmd_failed
1941 00009083 800D[7C930100]10    <1>       or     byte [AttributesMask], 10h ; ..directory..
1942 0000908A EB18                <1>       jmp    short get_dfname_fchar_attr
1943                              <1> pass_only_directories:
1944 0000908C 3C46                <1>       cmp    al, 'F'      ; files only ?
1945 0000908E 0F85B0000000        <1>         jne     check_attr_s
1946 00009094 AC                  <1>       lodsb
1947 00009095 3C20                <1>       cmp    al, 20h
1948 00009097 0F87E4FEFFFF        <1>         ja      loc_cmd_failed
1949 0000909D 800D[7D930100]10    <1>       or     byte [AttributesMask+1], 10h ; ..except directories..
1950                              <1> get_dfname_fchar_attr:
1951 000090A4 AC                  <1>       lodsb
1952 000090A5 3C20                <1>       cmp    al, 20h
1953 000090A7 74FB                <1>       je     short get_dfname_fchar_attr
1954 000090A9 725E                <1>       jb     short loc_print_dir_call_all
1955                              <1>
1956                              <1> loc_print_dir_call_flt:
1957 000090AB 4E                  <1>       dec    esi
1958 000090AC BF[7E930100]        <1>       mov    edi, FindFile_Drv
1959 000090B1 E8AC250000          <1>       call   parse_path_name
1960 000090B6 7308                <1>       jnc    short loc_print_dir_change_drv_1
1961 000090B8 3C01                <1>       cmp    al, 1
1962 000090BA 0F87ECFEFFFF        <1>         ja      loc_run_cmd_failed
1963                              <1>
1964                              <1> loc_print_dir_change_drv_1:
1965 000090C0 8A15[7E930100]      <1>       mov    dl, [FindFile_Drv]
1966                              <1> loc_print_dir_change_drv_2:
1967 000090C6 3A15[3A920100]      <1>       cmp    dl, [RUN_CDRV]
1968 000090CC 740B                <1>       je     short loc_print_dir_change_directory
1969 000090CE E81BF0FFFF          <1>       call   change_current_drive
1970 000090D3 0F82D3FEFFFF        <1>         jc      loc_run_cmd_failed
1971                              <1> loc_print_dir_change_directory:
1972 000090D9 803D[7F930100]20    <1>       cmp    byte [FindFile_Directory], 20h ; 0 or 20h ?
1973 000090E0 761D                <1>       jna    short pass_print_dir_change_directory
1974                              <1>
1975 000090E2 FE05[A6400100]      <1>       inc    byte [Restore_CDIR]
1976 000090E8 BE[7F930100]        <1>       mov    esi, FindFile_Directory
1977 000090ED 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
1978 000090EF E8581F0000          <1>       call   change_current_directory
1979 000090F4 0F82B2FEFFFF        <1>         jc      loc_run_cmd_failed
1980                              <1>
1981                              <1> loc_print_dir_change_prompt_dir_string:
1982 000090FA E86D1E0000          <1>       call   change_prompt_dir_string
1983                              <1>
1984                              <1> pass_print_dir_change_directory:
1985 000090FF BE[C0930100]        <1>       mov    esi, FindFile_Name
1986 00009104 803E20              <1>       cmp    byte [esi], 20h ; ; 0 or 20h ?
1987 00009107 7706                <1>       ja     short loc_print_dir_call
1988                              <1>
1989                              <1> loc_print_dir_call_all:
1990 00009109 C7062A2E2A00        <1>       mov    dword [esi], '*.*'
1991                              <1> loc_print_dir_call:
1992 0000910F E87E000000          <1>       call   print_directory
1993                              <1>
1994 00009114 8A15[3A920100]      <1>       mov    dl, [RUN_CDRV]  ; it is set at the beginning
1995 0000911A 3A15[DE8A0100]      <1>       cmp    dl, [Current_Drv]
1996 00009120 7406                <1>       je     short loc_print_dir_call_restore_cdir_retn
1997 00009122 E8C7EFFFFF          <1>       call   change_current_drive
1998 00009127 C3                  <1>       retn
1999                              <1>
2000                              <1> loc_print_dir_call_restore_cdir_retn:
2001 00009128 803D[A6400100]00    <1>       cmp    byte [Restore_CDIR], 0
2002 0000912F 7610                <1>       jna    short pass_print_dir_call_restore_cdir_retn
2003                              <1>
2004 00009131 BE00010900          <1>       mov    esi, Logical_DOSDisks
2005 00009136 31C0                <1>       xor    eax, eax
2006 00009138 88D4                <1>       mov    ah, dl
2007 0000913A 01C6                <1>       add    esi, eax
2008                              <1>
2009 0000913C E864F0FFFF          <1>       call   restore_current_directory
2010                              <1>
2011                              <1> pass_print_dir_call_restore_cdir_retn:
2012 00009141 C3                  <1>       retn
2013                              <1>
2014                              <1> check_attr_s_cap:
2015 00009142 24DF                <1>       and    al, 0DFh
2016                              <1> check_attr_s:
2017 00009144 3C53                <1>       cmp    al, 'S'
2018 00009146 7514                <1>       jne    short pass_attr_s
2019 00009148 800D[7C930100]04    <1>       or     byte [AttributesMask], 4 ; system
2020 0000914F AC                  <1>       lodsb
2021 00009150 3C20                <1>       cmp    al, 20h
2022 00009152 0F844CFFFFFF        <1>         je      get_dfname_fchar_attr
2023 00009158 72AF                <1>       jb     short loc_print_dir_call_all
2024 0000915A 24DF                <1>       and    al, 0DFh
2025                              <1> pass_attr_s:
2026 0000915C 3C48                <1>       cmp    al, 'H'
2027 0000915E 7514                <1>       jne    short pass_attr_h
2028 00009160 800D[7C930100]02    <1>       or     byte [AttributesMask], 2 ; hidden
2029                              <1> pass_attr_shr:
2030 00009167 AC                  <1>       lodsb
2031 00009168 3C20                <1>       cmp    al, 20h
2032 0000916A 0F8434FFFFFF        <1>         je      get_dfname_fchar_attr
2033 00009170 7297                <1>       jb     short loc_print_dir_call_all
2034 00009172 EBCE                <1>       jmp    short check_attr_s_cap
2035                              <1>
2036                              <1> pass_attr_h:
2037 00009174 3C52                <1>       cmp    al, 'R'
2038 00009176 7509                <1>       jne    short pass_attr_r
2039 00009178 800D[7C930100]01    <1>       or     byte [AttributesMask], 1 ; read only
2040 0000917F EBE6                <1>       jmp    short pass_attr_shr
2041                              <1>
2042                              <1> pass_attr_r:
```

```
2043 00009181 3C41                <1>       cmp    al, 'A'
2044 00009183 0F85F8FDFFFF        <1>         jne     loc_cmd_failed
2045 00009189 800D[7C930100]20    <1>       or     byte [AttributesMask], 20h ; archive
2046 00009190 EBD5                <1>       jmp    short pass_attr_shr
2047                              <1>
2048                              <1> print_directory:
2049                              <1>       ; 13/05/2016
2050                              <1>       ; 11/02/2016
2051                              <1>       ; 10/02/2016
2052                              <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
2053                              <1>       ; 30/10/2010 ('proc_print_directory')
2054                              <1>       ; 19/09/2009
2055                              <1>       ; 2005
2056                              <1>       ; INPUT ->
2057                              <1>       ;      ESI = Asciiz File/Dir Name Address
2058                              <1>
2059 00009192 56                  <1>       push   esi
2060                              <1>
2061 00009193 29C0                <1>       sub    eax, eax
2062                              <1>
2063 00009195 66A3[08940100]      <1>       mov    word [Dir_Count], ax ; 0
2064 0000919B 66A3[06940100]      <1>       mov    word [File_Count], ax ; 0
2065 000091A1 A3[0A940100]        <1>       mov    dword [Total_FSize], eax ; 0
2066                              <1>
2067 000091A6 E8D0E3FFFF          <1>       call    clear_screen
2068                              <1>
2069 000091AB 31C9                <1>       xor    ecx, ecx
2070 000091AD 8A2D[DE8A0100]      <1>       mov     ch, [Current_Drv] ; DirBuff_Drv - 'A'
2071 000091B3 A0[DF8A0100]        <1>       mov     al, [Current_Dir_Drv]
2072 000091B8 A2[63420100]        <1>       mov     [Dir_Drive_Name], al
2073 000091BD BE00010900          <1>       mov    esi, Logical_DOSDisks
2074 000091C2 01CE                <1>       add    esi, ecx
2075                              <1>
2076 000091C4 E858F9FFFF          <1>       call   move_volume_name_and_serial_no
2077 000091C9 730C                <1>       jnc    short print_dir_strlen_check
2078                              <1>
2079 000091CB 5E                  <1>       pop    esi
2080 000091CC 8A3D[468A0100]      <1>       mov    bh, [ptty] ; [ACTIVE_PAGE]
2081                              <1>       ;call  beeper
2082                              <1>       ;retn
2083 000091D2 E91A92FFFF          <1>       jmp    beeper  ; beep ! and return
2084                              <1>
2085                              <1> print_dir_strlen_check:
2086 000091D7 BE[E18A0100]        <1>       mov    esi, Current_Dir_Root
2087 000091DC BF[00430100]        <1>       mov    edi, Dir_Str_Root
2088                              <1>
2089                              <1>       ;xor   ecx, ecx
2090 000091E1 8A0D[3D8B0100]      <1>         mov     cl, [Current_Dir_StrLen]
2091 000091E7 FEC1                <1>       inc    cl
2092 000091E9 80F940              <1>       cmp    cl, 64
2093 000091EC 760D                <1>       jna    short pass_print_dir_strlen_shorting
2094 000091EE 46                  <1>       inc    esi
2095 000091EF 01CE                <1>       add    esi, ecx
2096 000091F1 83EE40              <1>       sub    esi, 64
2097 000091F4 47                  <1>       inc    edi
2098 000091F5 B82E2E2E20          <1>       mov    eax, '... '
2099 000091FA AB                  <1>       stosd
2100                              <1>
2101                              <1> pass_print_dir_strlen_shorting:
2102 000091FB F3A4                <1>       rep    movsb
2103                              <1>
2104 000091FD BE[56420100]        <1>       mov    esi, Dir_Drive_Str
2105 00009202 E85EE3FFFF          <1>       call   print_msg
2106                              <1>
2107 00009207 BE[B5420100]        <1>       mov    esi, Vol_Serial_Header
2108 0000920C E854E3FFFF          <1>       call   print_msg
2109                              <1>
2110 00009211 BE[F5420100]        <1>       mov    esi, Dir_Str_Header
2111 00009216 E84AE3FFFF          <1>       call   print_msg
2112                              <1>
2113 0000921B BE[414D0100]        <1>       mov    esi, next2line
2114 00009220 E840E3FFFF          <1>       call   print_msg
2115                              <1>
2116                              <1> loc_print_dir_first_file:
2117 00009225 C605[1D940100]10    <1>       mov    byte [PrintDir_RowCounter], 16
2118 0000922C 66A1[7C930100]      <1>       mov    ax, [AttributesMask]
2119 00009232 5E                  <1>       pop    esi
2120                              <1>
2121 00009233 E859020000          <1>       call   find_first_file
2122 00009238 0F826F010000        <1>         jc      loc_dir_ok
2123                              <1>
2124                              <1> loc_dfname_use_this:
2125                              <1>       ; bl = File Attributes (bh = Long Name Entry Length)
2126 0000923E F6C310              <1>       test   bl, 10h  ; Is it a directory?
2127 00009241 741B                <1>       jz     short loc_not_dir
2128                              <1>
2129 00009243 66FF05[08940100]    <1>       inc    word [Dir_Count]
2130 0000924A 89F2                <1>       mov    edx, esi     ; FindFile_DirEntry address
2131 0000924C BE[44440100]        <1>       mov    esi, Type_Dir; '<DIR>     '
2132 00009251 BF[5B440100]        <1>       mov    edi, Dir_Or_FileSize
2133                              <1>       ; move 10 bytes
2134 00009256 A5                  <1>       movsd
2135 00009257 A5                  <1>       movsd
2136 00009258 66A5                <1>       movsw
2137 0000925A 89D6                <1>       mov    esi, edx
2138 0000925C EB36                <1>       jmp     short loc_dir_attribute
2139                              <1>
2140                              <1> loc_not_dir:
2141 0000925E 66FF05[06940100]    <1>       inc    word [File_Count]
2142 00009265 0105[0A940100]      <1>       add    [Total_FSize], eax
2143                              <1>
2144 0000926B B90A000000          <1>       mov    ecx, 10  ; 32 bit divisor
2145 00009270 89CF                <1>       mov    edi, ecx
2146 00009272 81C7[5B440100]      <1>       add    edi, Dir_Or_FileSize
2147                              <1> loc_dir_rdivide:
```

```
2148 00009278 29D2                <1>       sub    edx, edx
2149 0000927A F7F1                <1>       div    ecx     ; remainder in dl (< 10)
2150 0000927C 80C230              <1>       add     dl, '0'      ; to make visible (ascii)
2151 0000927F 4F                  <1>       dec    edi
2152 00009280 8817                <1>       mov     [edi], dl
2153 00009282 21C0                <1>       and    eax, eax
2154 00009284 75F2                <1>       jnz    short loc_dir_rdivide
2155                              <1>
2156                              <1> loc_dir_fill_space:
2157 00009286 81FF[5B440100]      <1>       cmp     edi, Dir_Or_FileSize
2158 0000928C 7606                <1>       jna     short loc_dir_attribute
2159 0000928E 4F                  <1>       dec     edi
2160 0000928F C60720              <1>       mov     byte [edi], 20h
2161 00009292 EBF2                <1>       jmp     short loc_dir_fill_space
2162                              <1>
2163                              <1> loc_dir_attribute:
2164 00009294 C705[66440100]2020- <1>       mov    dword [File_Attribute], 20202020h
2164 0000929C 2020                <1>
2165                              <1>
2166 0000929E 80FB20              <1>       cmp    bl, 20h  ; Is it an archive file?
2167 000092A1 7207                <1>       jb     short loc_dir_pass_arch
2168 000092A3 C605[69440100]41    <1>       mov    byte [File_Attribute+3], 'A'
2169                              <1>
2170                              <1> loc_dir_pass_arch:
2171 000092AA 80E307              <1>       and    bl, 7
2172 000092AD 7428                <1>       jz     short loc_dir_file_name
2173 000092AF 88DF                <1>       mov    bh, bl
2174 000092B1 80E303              <1>       and    bl, 3
2175 000092B4 38DF                <1>       cmp    bh, bl
2176 000092B6 7607                <1>       jna    short loc_dir_pass_s
2177 000092B8 C605[66440100]53    <1>       mov    byte [File_Attribute], 'S'
2178                              <1>
2179                              <1> loc_dir_pass_s:
2180 000092BF 80E302              <1>       and     bl,2
2181 000092C2 7407                <1>       jz      short loc_dir_pass_h
2182 000092C4 C605[67440100]48    <1>       mov     byte [File_Attribute+1], 'H'
2183                              <1> loc_dir_pass_h:
2184 000092CB 80E701              <1>       and     bh,1
2185 000092CE 7407                <1>       jz      short loc_dir_file_name
2186 000092D0 C605[68440100]52    <1>       mov     byte [File_Attribute+2], 'R'
2187                              <1> loc_dir_file_name:
2188                              <1>       ;mov     bx, [esi+18h] ; Date
2189                              <1>       ;mov     dx, [esi+16h] ; Time
2190 000092D7 8B5E16              <1>       mov    ebx, [esi+16h]
2191 000092DA 89F1                <1>       mov    ecx, esi ; FindFile_DirEntry address
2192 000092DC BF[4E440100]        <1>       mov     edi, File_Name
2193                              <1>       ; move 8 bytes
2194 000092E1 A5                  <1>       movsd
2195 000092E2 A5                  <1>       movsd
2196 000092E3 C60720              <1>       mov    byte [edi], 20h
2197 000092E6 47                  <1>       inc    edi
2198                              <1>       ; move 3 bytes
2199 000092E7 66A5                <1>       movsw
2200 000092E9 A4                  <1>       movsb
2201 000092EA 89CE                <1>       mov    esi, ecx
2202                              <1>
2203                              <1> Dir_Time_start:
2204                              <1>       ;mov   ax, dx       ; Time
2205 000092EC 6689D8              <1>       mov    ax, bx
2206 000092EF 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
2207 000092F3 6683E03F            <1>       and    ax, 0000111111b     ; Minute Mask
2208 000092F7 D40A                <1>       aam                 ; Q([AL]/10)->AH
2209                              <1>                           ; R([AL]/10)->AL
2210                              <1>                           ; [AL]+[AH]= Minute as BCD
2211 000092F9 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
2212 000092FD 86E0                <1>       xchg   ah, al
2213 000092FF 66A3[79440100]      <1>       mov    [File_Minute], ax
2214                              <1>
2215                              <1>       ;mov   al, dh
2216 00009305 88F8                <1>       mov    al, bh
2217 00009307 C0E803              <1>       shr    al, 3        ; shift right 3 times
2218 0000930A D40A                <1>       aam                 ; [AL]+[AH]= Hours as BCD
2219 0000930C 660D3030            <1>       or     ax, '00'
2220 00009310 86E0                <1>       xchg   ah, al
2221 00009312 66A3[76440100]      <1>       mov     [File_Hour], ax
2222                              <1>
2223 00009318 C1EB10              <1>       shr    ebx, 16             ; BX = Date
2224                              <1>
2225                              <1> Dir_Date_start:
2226 0000931B 6689D8              <1>       mov    ax, bx       ; Date
2227 0000931E 6683E01F            <1>       and    ax, 00011111b; Day Mask
2228 00009322 D40A                <1>       aam                 ; Q([AL]/10)->AH
2229                              <1>                           ; R([AL]/10)->AL
2230                              <1>                           ; [AL]+[AH]= Day as BCD
2231 00009324 660D3030            <1>       or     ax, '00'     ; Convert to ASCII
2232 00009328 86C4                <1>       xchg   al, ah
2233                              <1>
2234 0000932A 66A3[6B440100]      <1>       mov    [File_Day], ax
2235                              <1>
2236 00009330 6689D8              <1>       mov    ax, bx
2237 00009333 66C1E805            <1>       shr    ax, 5        ; shift right 5 times
2238 00009337 6683E00F            <1>       and    ax, 00001111b; Month Mask
2239 0000933B D40A                <1>       aam
2240 0000933D 660D3030            <1>       or     ax, '00'
2241 00009341 86E0                <1>       xchg   ah, al
2242 00009343 66A3[6E440100]      <1>       mov    [File_Month], ax
2243                              <1>
2244 00009349 6689D8              <1>       mov    ax, bx
2245 0000934C 66C1E809            <1>       shr     ax, 9
2246 00009350 6683E07F            <1>       and    ax, 01111111b; Result = Year - 1980
2247 00009354 6605BC07            <1>       add    ax, 1980
2248                              <1>
2249 00009358 B10A                <1>       mov    cl, 10
2250 0000935A F6F1                <1>       div    cl           ; Q -> AL, R -> AH
2251 0000935C 80CC30              <1>       or     ah, '0'
```

```
0000935F 8825[74440100]      <1>       mov    [File_Year+3], ah
00009365 D40A                <1>       aam
00009367 86E0                <1>       xchg   ah, al
00009369 80CC30              <1>       or     ah, '0'        ; Convert to ASCII
0000936C 8825[73440100]      <1>       mov    [File_Year+2], ah
00009372 D40A                <1>       aam
00009374 86C4                <1>       xchg   al, ah
00009376 660D3030            <1>       or     ax, '00'
0000937A 66A3[71440100]      <1>       mov    [File_Year], ax
                             <1>
                             <1> loc_show_line:
00009380 56                  <1>       push   esi
00009381 BE[4E440100]        <1>       mov     esi, File_Name
00009386 E8DAE1FFFF          <1>       call   print_msg
0000938B BE[434D0100]        <1>       mov    esi, nextline
00009390 E8D0E1FFFF          <1>       call   print_msg
00009395 5E                  <1>       pop    esi
                             <1>
00009396 FE0D[1D940100]      <1>       dec    byte [PrintDir_RowCounter]
0000939C 0F84D4000000        <1>         jz      pause_dir_scroll
                             <1>
                             <1> loc_next_entry:
000093A2 E899010000          <1>       call   find_next_file
000093A7 0F8391FEFFFF        <1>         jnc     loc_dfname_use_this
                             <1>
                             <1> loc_dir_ok:
000093AD B90A000000          <1>       mov     ecx, 10
000093B2 66A1[08940100]      <1>       mov    ax, [Dir_Count]
000093B8 BF[8F440100]        <1>       mov    edi, Decimal_Dir_Count
000093BD 6639C8              <1>       cmp    ax, cx ; 10
000093C0 7216                <1>       jb     short pass_ddc
000093C2 47                  <1>       inc    edi
000093C3 6683F864            <1>       cmp    ax, 100
000093C7 720F                <1>       jb     short pass_ddc
000093C9 47                  <1>       inc    edi
000093CA 663DE803            <1>       cmp    ax, 1000
000093CE 7208                <1>       jb     short pass_ddc
000093D0 47                  <1>       inc    edi
000093D1 663D1027            <1>       cmp    ax, 10000
000093D5 7201                <1>       jb     short pass_ddc
000093D7 47                  <1>       inc    edi
                             <1> pass_ddc:
000093D8 886F01              <1>       mov     [edi+1], ch ; 0
                             <1> loc_ddc_rediv:
000093DB 31D2                <1>       xor     edx, edx
000093DD 66F7F1              <1>       div     cx   ; 10
000093E0 80C230              <1>       add     dl, '0'
000093E3 8817                <1>       mov     [edi], dl
000093E5 4F                  <1>       dec     edi
000093E6 6609C0              <1>       or     ax, ax
000093E9 75F0                <1>       jnz    short loc_ddc_rediv
                             <1>
000093EB 66A1[06940100]      <1>       mov     ax, [File_Count]
000093F1 BF[7E440100]        <1>       mov     edi, Decimal_File_Count
000093F6 6639C8              <1>       cmp     ax, cx ; 10
000093F9 7216                <1>       jb      short pass_dfc
000093FB 47                  <1>       inc     edi
000093FC 6683F864            <1>       cmp     ax, 100
00009400 720F                <1>       jb      short pass_dfc
00009402 47                  <1>       inc     edi
00009403 663DE803            <1>       cmp     ax, 1000
00009407 7208                <1>       jb      short pass_dfc
00009409 47                  <1>       inc     edi
0000940A 663D1027            <1>       cmp     ax, 10000
0000940E 7201                <1>       jb      short pass_dfc
00009410 47                  <1>       inc     edi
                             <1> pass_dfc:
                             <1>       ;mov    cx, 10
00009411 886F01              <1>       mov     [edi+1], ch ; 00
                             <1> loc_dfc_rediv:
                             <1>       ;xor   dx, dx
00009414 30D2                <1>       xor    dl, dl
00009416 66F7F1              <1>       div    cx
00009419 80C230              <1>       add    dl, '0'
0000941C 8817                <1>       mov    [edi], dl
0000941E 4F                  <1>       dec    edi
0000941F 6609C0              <1>       or     ax, ax
00009422 75F0                <1>       jnz    short loc_dfc_rediv
                             <1>
00009424 BF[1C940100]        <1>       mov     edi, TFS_Dec_End
                             <1>         ;mov    byte [edi], 0
00009429 A1[0A940100]        <1>       mov     eax, [Total_FSize]
                             <1>       ;mov    ecx, 10
                             <1> rediv_tfs_hex:
                             <1>       ;sub   edx, edx
0000942E 28D2                <1>       sub    dl, dl
00009430 F7F1                <1>       div    ecx
00009432 80C230              <1>       add    dl, '0'
00009435 4F                  <1>       dec     edi
00009436 8817                <1>       mov     [edi], dl
00009438 21C0                <1>       and    eax, eax
0000943A 75F2                <1>       jnz    short rediv_tfs_hex
                             <1>
0000943C 893D[0E940100]      <1>       mov    [TFS_Dec_Begin], edi
00009442 BE[7C440100]        <1>       mov    esi, Decimal_File_Count_Header
00009447 E819E1FFFF          <1>       call   print_msg
0000944C BE[84440100]        <1>       mov    esi, str_files
00009451 E80FE1FFFF          <1>       call   print_msg
00009456 BE[95440100]        <1>       mov    esi, str_dirs
0000945B E805E1FFFF          <1>       call   print_msg
00009460 8B35[0E940100]      <1>       mov    esi, [TFS_Dec_Begin]
00009466 E8FAE0FFFF          <1>       call   print_msg
0000946B BE[A6440100]        <1>       mov    esi, str_bytes
00009470 E8F0E0FFFF          <1>       call   print_msg
                             <1>
```

```
2357 00009475 C3                  <1>       retn
2358                              <1>
2359                              <1> pause_dir_scroll:
2360 00009476 28E4                <1>       sub    ah, ah
2361 00009478 E8707AFFFF          <1>       call   int16h
2362 0000947D 3C1B                <1>       cmp    al, 1Bh
2363 0000947F 0F8428FFFFFF        <1>         je      loc_dir_ok
2364 00009485 C605[1D940100]10    <1>       mov    byte [PrintDir_RowCounter], 16 ; Reset counter
2365 0000948C E911FFFFFF          <1>         jmp     loc_next_entry
2366                              <1>
2367                              <1> find_first_file:
2368                              <1>       ; 11/02/2016
2369                              <1>       ; 10/02/2016
2370                              <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
2371                              <1>       ; 09/10/2011
2372                              <1>       ; 17/09/2009
2373                              <1>       ; 2005
2374                              <1>       ; INPUT ->
2375                              <1>       ;      ESI = ASCIIZ File/Dir Name Address (in Current Directory)
2376                              <1>       ;      AL = Attributes AND mask (The AND result must be equal to AL)
2377                              <1>       ;            bit 0 = Read Only
2378                              <1>       ;            bir 1 = Hidden
2379                              <1>       ;            bit 2 = System
2380                              <1>       ;            bit 3 = Volume Label
2381                              <1>       ;            bit 4 = Directory
2382                              <1>       ;            bit 5 = Archive
2383                              <1>       ;            bit 6 = Reserved, must be 0
2384                              <1>       ;            bit 7 = Reserved, must be 0
2385                              <1>       ;       AH = Attributes Negative AND mask (The AND result must be ZERO)
2386                              <1>       ;
2387                              <1>       ; OUTPUT ->
2388                              <1>       ;      CF = 1 -> Error, Error Code in EAX (AL)
2389                              <1>       ;      CF = 0 ->
2390                              <1>       ;           ESI = Directory Entry (FindFile_DirEntry) Location
2391                              <1>       ;           EDI = Directory Buffer Directory Entry Location
2392                              <1>       ;           EAX = File Size
2393                              <1>       ;            BL = Attributes of The File/Directory
2394                              <1>       ;            BH = Long Name Yes/No Status (>0 is YES)
2395                              <1>       ;              DX > 0 : Ambiguous filename chars are used
2396                              <1>       ;
2397                              <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
2398                              <1>
2399 00009491 66A3[CE930100]      <1>       mov    [FindFile_AttributesMask], ax
2400 00009497 BF[D0930100]        <1>       mov    edi, FindFile_DirEntry ; TR-DOS Fullfilename formatted buffer
2401 0000949C 31C0                <1>       xor    eax, eax
2402 0000949E B90B000000          <1>       mov    ecx, 11
2403 000094A3 F3AB                <1>       rep    stosd  ; 44 bytes
2404                              <1>       ;stosw       ; +2 bytes
2405                              <1>
2406 000094A5 BF[C0930100]        <1>       mov    edi, FindFile_Name ; FFF structure, offset 66
2407 000094AA 39FE                <1>       cmp    esi, edi
2408 000094AC 7408                <1>       je     short loc_fff_mfn_ok
2409 000094AE 89FA                <1>       mov    edx, edi
2410                              <1>        ; move 13 bytes
2411 000094B0 A5                  <1>       movsd
2412 000094B1 A5                  <1>       movsd
2413 000094B2 A5                  <1>       movsd
2414 000094B3 AA                  <1>       stosb
2415 000094B4 89D6                <1>       mov    esi, edx
2416                              <1> loc_fff_mfn_ok:
2417 000094B6 BF[6F930100]        <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
2418 000094BB E8D7200000          <1>       call   convert_file_name
2419 000094C0 89FE                <1>       mov    esi, edi ; offset Dir_Entry_Name
2420                              <1>
2421 000094C2 66A1[CE930100]      <1>       mov    ax, [FindFile_AttributesMask]
2422                              <1>       ;xor   ecx, ecx
2423 000094C8 30C9                <1>       xor    cl, cl
2424 000094CA E8D01D0000          <1>       call   locate_current_dir_file
2425 000094CF 726E                <1>       jc     short loc_fff_retn
2426                              <1>       ; EDI = Directory Entry
2427                              <1>       ; EBX = Directory Buffer Entry Index/Number
2428                              <1>
2429                              <1> loc_fff_fnf_ln_check:
2430 000094D1 30ED                <1>       xor    ch, ch
2431 000094D3 80F60F              <1>       xor    dh, 0Fh
2432 000094D6 7408                <1>       jz     short loc_fff_longname_yes
2433 000094D8 882D[CD930100]      <1>       mov    [FindFile_LongNameYes], ch ; 0
2434 000094DE EB0C                <1>       jmp    short loc_fff_longname_no
2435                              <1>
2436                              <1> loc_fff_longname_yes:
2437                              <1>       ;inc   byte [FindFile_LongNameYes]
2438 000094E0 8A0D[DA920100]      <1>       mov    cl, [LFN_EntryLength]
2439 000094E6 880D[CD930100]      <1>       mov    [FindFile_LongNameEntryLength], cl ; FindFile_LongNameYes
2440                              <1>
2441                              <1> loc_fff_longname_no:
2442                              <1>       ;mov   bx, [DirBuff_CurrentEntry]
2443 000094EC 66891D[F8930100]    <1>       mov    [FindFile_DirEntryNumber], bx
2444 000094F3 6689C2              <1>       mov    dx, ax ; Ambiguous Filename chars used sign > 0
2445                              <1>
2446 000094F6 A0[DE8A0100]        <1>       mov    al, [Current_Drv]
2447 000094FB A2[7E930100]        <1>       mov    [FindFile_Drv], al
2448                              <1>
2449 00009500 A1[D88A0100]        <1>       mov    eax, [Current_Dir_FCluster]
2450 00009505 A3[F0930100]        <1>       mov    [FindFile_DirFirstCluster], eax
2451                              <1>
2452 0000950A A1[09920100]        <1>       mov    eax, [DirBuff_Cluster]
2453 0000950F A3[F4930100]        <1>       mov    [FindFile_DirCluster], eax
2454                              <1>
2455 00009514 66FF05[FA930100]    <1>       inc    word [FindFile_MatchCounter]
2456                              <1>
2457 0000951B 89FB                <1>       mov    ebx, edi
2458 0000951D 89FE                <1>       mov    esi, edi
2459 0000951F BF[D0930100]        <1>       mov    edi, FindFile_DirEntry
2460 00009524 89F8                <1>       mov    eax, edi
2461 00009526 B108                <1>       mov    cl, 8
```

```
2462 00009528 F3A5               <1>       rep    movsd
2463 0000952A 89C6               <1>       mov    esi, eax
2464 0000952C 89DF               <1>       mov    edi, ebx
2465                             <1>
2466 0000952E A1[EC930100]       <1>       mov    eax, [FindFile_DirEntry+28] ; File Size
2467                             <1>
2468 00009533 8A1D[DB930100]     <1>       mov    bl, [FindFile_DirEntry+11] ; File Attributes
2469 00009539 8A3D[CD930100]     <1>       mov    bh, [FindFile_LongNameYes]
2470                             <1>
2471                             <1>       ;mov   cx, [DirBuff_EntryCounter]
2472                             <1>       ;mov   [FindFile_DirEntryNumber], cx
2473                             <1>       ;mov   cx, [FindFile_DirEntryNumber]
2474                             <1>       ; ecx = 0
2475                             <1>
2476                             <1> loc_fff_retn:
2477 0000953F C3                 <1>       retn
2478                             <1>
2479                             <1> find_next_file:
2480                             <1>       ; 15/10/2016
2481                             <1>       ; 10/02/2016
2482                             <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
2483                             <1>       ; 06/02/2011
2484                             <1>       ; 17/09/2009
2485                             <1>       ; 2005
2486                             <1>       ; INPUT ->
2487                             <1>       ;      NONE, Find First File Parameters
2488                             <1>       ; OUTPUT ->
2489                             <1>       ;      CF = 1 -> Error, Error Code in EAX (AL)
2490                             <1>       ;      CF = 0 ->
2491                             <1>       ;          ESI = Directory Entry (FindFile_DirEntry) Location
2492                             <1>       ;          EDI = Directory Buffer Directory Entry Location
2493                             <1>       ;          EAX = File Size
2494                             <1>       ;            BL = Attributes of The File/Directory
2495                             <1>       ;            BH = Long Name Yes/No Status (>0 is YES)
2496                             <1>       ;             DX > 0 : Ambiguous filename chars are used
2497                             <1>       ;
2498                             <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
2499                             <1>
2500 00009540 66833D[FA930100]00 <1>       cmp    word [FindFile_MatchCounter], 0
2501 00009548 7707               <1>       ja     short loc_start_search_next_file
2502                             <1>
2503                             <1> loc_fnf_stc_retn:
2504 0000954A F9                 <1>       stc
2505                             <1> loc_fnf_ax12h_retn:
2506 0000954B B80C000000         <1>       mov    eax, 12 ; No More files
2507                             <1> ;loc_fnf_retn:
2508 00009550 C3                 <1>       retn
2509                             <1>
2510                             <1> loc_start_search_next_file:
2511 00009551 668B1D[F8930100]   <1>       mov    bx, [FindFile_DirEntryNumber]
2512 00009558 6643               <1>       inc    bx
2513 0000955A 663B1D[07920100]   <1>       cmp    bx, [DirBuff_LastEntry]
2514 00009561 7719               <1>       ja     short loc_cont_search_next_file
2515                             <1>
2516                             <1> loc_fnf_search:
2517 00009563 BE[6F930100]       <1>       mov    esi, Dir_Entry_Name
2518 00009568 66A1[CE930100]     <1>       mov    ax, [FindFile_AttributesMask]
2519 0000956E 6631C9             <1>       xor    cx, cx
2520 00009571 E82D1E0000         <1>       call   find_directory_entry
2521 00009576 0F8355FFFFFF       <1>         jnc     loc_fff_fnf_ln_check
2522                             <1>
2523                             <1> loc_cont_search_next_file:
2524 0000957C 31DB               <1>       xor    ebx, ebx
2525 0000957E 8A3D[DE8A0100]     <1>       mov    bh, [Current_Drv]
2526 00009584 BE00010900         <1>       mov    esi, Logical_DOSDisks
2527 00009589 01DE               <1>       add    esi, ebx
2528                             <1>
2529 0000958B 803D[DC8A0100]00   <1>       cmp    byte [Current_Dir_Level], 0
2530 00009592 7608               <1>       jna    short loc_fnf_check_FAT_type
2531 00009594 807E0301           <1>       cmp    byte [esi+LD_FATType], 1
2532 00009598 72B1               <1>       jb     short loc_fnf_ax12h_retn
2533 0000959A EB06               <1>       jmp    short loc_fnf_check_next_cluster
2534                             <1>
2535                             <1> loc_fnf_check_FAT_type:
2536 0000959C 807E0303           <1>       cmp    byte [esi+LD_FATType], 3
2537 000095A0 72A9               <1>       jb     short loc_fnf_ax12h_retn
2538                             <1>
2539                             <1> loc_fnf_check_next_cluster:
2540 000095A2 A1[09920100]       <1>       mov    eax, [DirBuff_Cluster]
2541 000095A7 E8CA370000         <1>       call   get_next_cluster
2542 000095AC 7306               <1>       jnc    short loc_fnf_load_next_dir_cluster
2543 000095AE 09C0               <1>       or     eax, eax
2544 000095B0 7498               <1>       jz     short loc_fnf_stc_retn
2545                             <1>       ;mov   eax, 17 ;Drive not ready or read error
2546 000095B2 F5                 <1>       cmc    ;stc
2547                             <1> loc_fnf_retn:
2548 000095B3 C3                 <1>       retn
2549                             <1>
2550                             <1> loc_fnf_load_next_dir_cluster:
2551 000095B4 E8A3390000         <1>       call   load_FAT_sub_directory
2552 000095B9 72F8               <1>       jc     short loc_fnf_retn
2553 000095BB 6631DB             <1>       xor    bx, bx
2554 000095BE 66891D[F8930100]   <1>       mov    [FindFile_DirEntryNumber], bx
2555 000095C5 EB9C               <1>       jmp    short loc_fnf_search
2556                             <1>
2557                             <1> get_and_print_longname:
2558                             <1>       ; 16/10/2016
2559                             <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
2560                             <1>       ; 24/01/2010
2561                             <1>       ; 17/10/2009 (CMD_INTR.ASM, 'cmp_cmd_longname')
2562                             <1> get_longname_fchar:
2563 000095C7 803E20             <1>       cmp    byte [esi], 20h
2564 000095CA 7701               <1>       ja     short loc_find_longname
2565                             <1>       ;jb    short loc_longname_retn
2566                             <1>       ;inc   esi
```

```
                        <1>       ;je    short get_longname_fchar
                        <1> ;loc_longname_retn:
000095CC C3             <1>       retn
                        <1> loc_find_longname:
000095CD E839210000      <1>       call   find_longname
000095D2 7328           <1>       jnc    short loc_print_longname
                        <1>
000095D4 08C0           <1>       or     al, al
000095D6 741A           <1>       jz     short loc_longname_not_found
                        <1>
                        <1>       ; 16/10/2016 (15h -> 15, 17)
000095D8 3C0F           <1>       cmp    al, 15
000095DA 0F84B6F7FFFF    <1>       je     cd_drive_not_ready ; drive not ready
                        <1>                                 ; or
000095E0 3C11           <1>       cmp    al, 17          ; read error
000095E2 0F84AEF7FFFF    <1>       je     cd_drive_not_ready
                        <1>
                        <1> loc_ln_file_dir_not_found:
000095E8 BE[D1430100]    <1>       mov    esi, Msg_File_Directory_Not_Found
                        <1>       ;call  print_msg
                        <1>         ;retn
000095ED E973DFFFFF      <1>       jmp    print_msg
                        <1>
                        <1> loc_longname_not_found:
000095F2 BE[F0430100]    <1>         mov     esi, Msg_LongName_Not_Found
                        <1>       ;call  print_msg
                        <1>         ;retn
000095F7 E969DFFFFF      <1>       jmp    print_msg
                        <1>
                        <1> loc_print_longname:
                        <1>       ;mov   esi, LongFileName
000095FC BF[DE8B0100]    <1>       mov    edi, TextBuffer
00009601 57             <1>       push   edi
00009602 3C00           <1>       cmp    al, 0
00009604 7708           <1>       ja     short loc_print_longname_1
                        <1> loc_print_FS_longname: ; Singlix FS (64 byte ASCIIZ file name)
00009606 AC             <1>       lodsb
00009607 AA             <1>       stosb
00009608 08C0           <1>       or     al, al
0000960A 75FA           <1>       jnz    short loc_print_FS_longname
0000960C EB07           <1>       jmp    short loc_print_longname_2
                        <1>       ;
                        <1> loc_print_longname_1: ; MS Windows long name (UNICODE chars)
0000960E 66AD           <1>       lodsw
00009610 AA             <1>       stosb
00009611 08C0           <1>       or     al, al
00009613 75F9           <1>       jnz    short loc_print_longname_1
                        <1>       ;
                        <1> loc_print_longname_2:
00009615 5E             <1>       pop    esi
00009616 E84ADFFFFF      <1>       call   print_msg
0000961B BE[434D0100]    <1>       mov    esi, nextline
                        <1>       ;call  print_msg
                        <1>       ;retn
00009620 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:
00009625 BF[7E930100]    <1>       mov    edi, FindFile_Drv
0000962A E833200000      <1>       call   parse_path_name
0000962F 0F824CF9FFFF    <1>       jc     loc_cmd_failed
                        <1>
                        <1> loc_show_check_filename_exists:
00009635 BE[C0930100]    <1>       mov    esi, FindFile_Name
0000963A 803E20         <1>       cmp    byte [esi], 20h
0000963D 0F863EF9FFFF    <1>       jna    loc_cmd_failed
                        <1>
                        <1>       ; 15/02/2016 (invalid file name check)
00009643 E807020000      <1>       call   check_filename
00009648 730A           <1>       jnc    short loc_show_change_drv
                        <1>
0000964A BE[BC440100]    <1>       mov    esi, Msg_invalid_name_chars
0000964F E911DFFFFF      <1>       jmp    print_msg
                        <1>
                        <1> loc_show_change_drv:
00009654 8A35[DE8A0100]  <1>       mov    dh, [Current_Drv]
0000965A 8835[3A920100]  <1>       mov    [RUN_CDRV], dh
00009660 8A15[7E930100]  <1>       mov    dl, [FindFile_Drv]
00009666 38F2           <1>       cmp    dl, dh
00009668 740B           <1>       je     short loc_show_change_directory
0000966A E87FEAFFFF      <1>       call   change_current_drive
                        <1>       ;jc    loc_file_rw_cmd_failed
0000966F 0F8237F9FFFF    <1>       jc     loc_run_cmd_failed
                        <1>
                        <1> loc_show_change_directory:
00009675 803D[7F930100]20 <1>       cmp    byte [FindFile_Directory], 20h
0000967C 7618           <1>       jna    short loc_findload_showfile
                        <1>
0000967E FE05[A6400100]  <1>       inc    byte [Restore_CDIR]
00009684 BE[7F930100]    <1>       mov    esi, FindFile_Directory
00009689 30E4           <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000968B E8BC190000      <1>       call   change_current_directory
                        <1>       ;jc    loc_file_rw_cmd_failed
00009690 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:
```

```
2672                                    <1>       ; 15/02/2016
2673 00009696 BE[C0930100]              <1>       mov    esi, FindFile_Name
2674 0000969B BF[6F930100]              <1>       mov    edi, Dir_Entry_Name ; Dir Entry Format File Name
2675 000096A0 E8F21E0000                <1>       call   convert_file_name
2676 000096A5 89FE                      <1>       mov    esi, edi ; offset Dir_Entry_Name
2677                                    <1>
2678 000096A7 28C0                      <1>       sub    al, al ; Attrib AND mask = 0
2679                                    <1>       ; Directory attribute : 10h
2680                                    <1>       ; Volume name attribute: 8h
2681 000096A9 B418                      <1>       mov    ah, 00011000b ; 18h (Attrib NAND, AND --> zero mask)
2682                                    <1>       ;
2683 000096AB 6631C9                    <1>       xor    cx, cx
2684 000096AE E8EC1B0000                <1>       call   locate_current_dir_file
2685                                    <1>       ;jc    loc_file_rw_cmd_failed
2686 000096B3 0F82F3F8FFFF              <1>       jc     loc_run_cmd_failed
2687                                    <1>
2688                                    <1> loc_show_load_file:
2689                                    <1>       ; EDI = Directory Entry
2690 000096B9 668B4714                  <1>       mov    ax, [edi+DirEntry_FstClusHI] ; First Cluster High Word
2691 000096BD C1E010                    <1>       shl    eax, 16
2692 000096C0 668B471A                  <1>       mov    ax, [edi+DirEntry_FstClusLO] ; First Cluster Low Word
2693 000096C4 A3[28940100]              <1>       mov    [Show_Cluster], eax
2694 000096C9 8B471C                    <1>       mov    eax, [edi+DirEntry_FileSize] ; File Size
2695 000096CC 21C0                      <1>       and    eax, eax ; Empty file !
2696 000096CE 0F8491000000              <1>         jz     end_of_show_file
2697 000096D4 A3[2C940100]              <1>       mov    [Show_FileSize], eax
2698 000096D9 31C0                      <1>       xor    eax, eax
2699 000096DB A3[30940100]              <1>       mov    [Show_FilePointer], eax ; 0
2700 000096E0 66A3[34940100]            <1>       mov    [Show_ClusterPointer], ax ; 0
2701 000096E6 29DB                      <1>       sub    ebx, ebx
2702 000096E8 8A3D[DE8A0100]            <1>       mov    bh, [Current_Drv]
2703 000096EE BE00010900                <1>       mov    esi, Logical_DOSDisks
2704 000096F3 01DE                      <1>       add    esi, ebx
2705 000096F5 8935[24940100]            <1>       mov    [Show_LDDDT], esi ; Logical DOS Drv Description Table addr
2706                                    <1>
2707 000096FB 807E0300                  <1>       cmp    byte [esi+LD_FATType], 0
2708 000096FF 7713                      <1>       ja     short loc_show_calculate_cluster_size
2709                                    <1>       ; Singlix FS
2710                                    <1>       ; First Cluster Number is FDT number (in compatibility buffer)
2711 00009701 8B15[28940100]            <1>       mov    edx, [Show_Cluster] ; Compatibility dir. buffer value (FDT)
2712 00009707 8915[20940100]            <1>       mov    [Show_FDT], edx
2713 0000970D 31C0                      <1>       xor    eax, eax
2714 0000970F A3[28940100]              <1>       mov    [Show_Cluster], eax ; Sector index  = 0
2715                                    <1>                                ; (next time it will be 1)
2716                                    <1> loc_show_calculate_cluster_size:
2717 00009714 668B5E11                  <1>       mov    bx, [esi+LD_BPB+BPB_BytsPerSec] ; FAT 12-16-32 (512)
2718                                    <1>       ; BX = 512 = [esi+LD_FS_BytesPerSec] ; Singlix FS
2719 00009718 8A4613                    <1>       mov    al, [esi+LD_BPB+BPB_SecPerClust] ; FAT 12-16-32 (<= 128)
2720                                    <1>       ; AL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
2721 0000971B F7E3                      <1>       mul    ebx
2722                                    <1>
2723                                    <1>       ;cmp   eax, 65536 ; non-compatible (very big) cluster size
2724                                    <1>       ;ja    short end_of_show_file
2725 0000971D 66A3[36940100]            <1>       mov    [Show_ClusterSize], ax
2726                                    <1>
2727                                    <1> loc_start_show_file:
2728 00009723 BE[434D0100]              <1>       mov    esi, nextline
2729 00009728 E838DEFFFF                <1>       call   print_msg
2730                                    <1>
2731 0000972D A1[28940100]              <1>       mov    eax, [Show_Cluster]
2732 00009732 C605[38940100]17          <1>       mov    byte [Show_RowCount], 23
2733                                    <1>
2734                                    <1>       ; 17/02/2016
2735 00009739 8B35[24940100]            <1>       mov    esi, [Show_LDDDT]
2736                                    <1>
2737                                    <1> loc_show_next_cluster:
2738                                    <1>       ; 15/02/2016
2739 0000973F BB00000700                <1>       mov    ebx, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
2740                                    <1>       ; ESI = Logical DOS drv description table address
2741 00009744 E851380000                <1>       call   read_cluster
2742                                    <1>       ;jc    loc_file_rw_cmd_failed
2743 00009749 0F825DF8FFFF              <1>       jc     loc_run_cmd_failed
2744                                    <1>
2745 0000974F 31DB                      <1>       xor    ebx, ebx
2746                                    <1> loc_show_next_byte:
2747 00009751 803D[38940100]00          <1>       cmp    byte [Show_RowCount], 0
2748 00009758 7521                      <1>       jne    short pass_show_wait_for_key
2749 0000975A 30E4                      <1>       xor    ah, ah
2750 0000975C E88C77FFFF                <1>       call   int16h
2751 00009761 3C1B                      <1>       cmp    al, 1Bh
2752 00009763 750F                      <1>       jne    short pass_exit_show
2753                                    <1> end_of_show_file:
2754                                    <1> pass_show_file:
2755 00009765 BE[434D0100]              <1>       mov    esi, nextline
2756 0000976A E8F6DDFFFF                <1>       call   print_msg
2757 0000976F E94B010000                <1>       jmp    loc_file_rw_restore_retn
2758                                    <1>
2759                                    <1> pass_exit_show:
2760 00009774 C605[38940100]14          <1>       mov    byte [Show_RowCount], 20
2761                                    <1> pass_show_wait_for_key:
2762 0000977B 81C300000700              <1>       add    ebx, Cluster_Buffer
2763 00009781 8A03                      <1>       mov    al, [ebx]
2764 00009783 3C0D                      <1>       cmp    al, 0Dh
2765 00009785 0F8590000000              <1>         jne    loc_show_check_tab_space
2766 0000978B FE0D[38940100]            <1>       dec    byte [Show_RowCount]
2767                                    <1> pass_show_dec_rowcount:
2768 00009791 B307                      <1>       mov    bl, 7 ; (light gray character color, black background)
2769 00009793 8A3D[468A0100]            <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
2770 00009799 E8678BFFFF                <1>       call   _write_tty
2771                                    <1> loc_show_check_eof:
2772 0000979E FF05[30940100]            <1>       inc    dword [Show_FilePointer]
2773 000097A4 A1[30940100]              <1>       mov    eax, [Show_FilePointer]
2774 000097A9 3B05[2C940100]            <1>       cmp    eax, [Show_FileSize]
2775 000097AF 73B4                      <1>       jnb    short end_of_show_file
2776 000097B1 66FF05[34940100]          <1>       inc    word [Show_ClusterPointer]
```

```
2777 000097B8 0FB71D[34940100]    <1>       movzx  ebx, word [Show_ClusterPointer]
2778                              <1>
2779                              <1>       ; 17/02/2016
2780                              <1>       ; (sector boundary -9 bits- check, 512 = 0)
2781 000097BF 66F7C3FF01          <1>         test    bx, 1FFh ;  1 to 511
2782 000097C4 758B                <1>       jnz    short loc_show_next_byte
2783                              <1>
2784                              <1>       ; 16/02/2016
2785 000097C6 8B35[24940100]      <1>       mov    esi, [Show_LDDDT]
2786                              <1>       ;
2787 000097CC 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
2788 000097D0 7719                <1>       ja     short loc_show_check_fat_cluster_size
2789                              <1>
2790                              <1>       ; Singlix FS
2791                              <1>       ; 1 sector, more... (cluster size = 1 sector)
2792 000097D2 A1[28940100]        <1>       mov    eax, [Show_Cluster]
2793 000097D7 40                  <1>       inc    eax
2794 000097D8 A3[28940100]        <1>       mov    [Show_Cluster], eax
2795                              <1>
2796 000097DD 6621DB              <1>       and    bx, bx ; 65536 -> 0
2797 000097E0 0F856BFFFFFF        <1>         jnz  loc_show_next_byte
2798 000097E6 E954FFFFFF          <1>       jmp     loc_show_next_cluster
2799                              <1>
2800                              <1> loc_show_check_fat_cluster_size:
2801                              <1>       ; 17/02/2016
2802 000097EB 663B1D[36940100]    <1>       cmp    bx, [Show_ClusterSize] ; cluster size in bytes
2803 000097F2 0F8259FFFFFF        <1>         jb   loc_show_next_byte
2804 000097F8 66C705[34940100]00- <1>       mov    word [Show_ClusterPointer], 0
2804 00009800 00                  <1>
2805                              <1>
2806 00009801 A1[28940100]        <1>       mov    eax, [Show_Cluster]
2807                              <1>       ;mov   esi, [Show_LDDDT]
2808                              <1> loc_show_get_next_cluster:
2809 00009806 E86B350000          <1>       call   get_next_cluster
2810                              <1>       ;jc    loc_file_rw_cmd_failed
2811 0000980B 0F829BF7FFFF        <1>       jc     loc_run_cmd_failed
2812                              <1> loc_show_update_ccluster:
2813 00009811 A3[28940100]        <1>       mov    [Show_Cluster], eax
2814 00009816 E924FFFFFF          <1>         jmp     loc_show_next_cluster
2815                              <1>
2816                              <1> loc_show_check_tab_space:
2817 0000981B 3C09                <1>       cmp    al, 09h
2818 0000981D 0F856EFFFFFF        <1>         jne     pass_show_dec_rowcount
2819                              <1> loc_show_put_tab_space:
2820 00009823 8A3D[468A0100]      <1>       mov    bh, [ACTIVE_PAGE] ; [ptty]
2821 00009829 E85387FFFF          <1>       call   get_cpos
2822                              <1>       ; dl = cursor column
2823 0000982E 80E207              <1>       and    dl, 7 ; 18/02/2016
2824                              <1>       ;shr   bh, 1 ; [ACTIVE_PAGE]
2825 00009831 8A3D[468A0100]      <1>       mov    bh, [ACTIVE_PAGE]
2826 00009837 B307                <1>       mov    bl, 7 ; color attribute
2827                              <1> loc_show_put_space_chars:
2828 00009839 B020                <1>       mov    al, 20h ; space
2829                              <1>       ;mov   bh, [ACTIVE_PAGE] ; [ptty]
2830                              <1>       ;mov   bl, 7 ; color attribute
2831                              <1>       ;push  dx
2832 0000983B 52                  <1>       push   edx ; 29/12/2017
2833 0000983C E8C48AFFFF          <1>       call   _write_tty
2834 00009841 5A                  <1>       pop    edx ; 29/12/2017
2835                              <1>       ;pop   dx
2836                              <1>       ; 18/02/2016
2837 00009842 80FA07              <1>       cmp    dl, 7
2838 00009845 0F8353FFFFFF        <1>       jnb    loc_show_check_eof
2839 0000984B FEC2                <1>       inc    dl
2840 0000984D EBEA                <1>       jmp    short loc_show_put_space_chars
2841                              <1>
2842                              <1> check_filename:
2843                              <1>       ; 10/10/2016
2844                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
2845                              <1>       ; 07/08/2010 (FILE.ASM, 'proc_check_filename')
2846                              <1>       ; 10/07/2010
2847                              <1>       ; Derived from 'proc_check_filename'
2848                              <1>       ; in the old TRDOS.ASM (09/02/2005).
2849                              <1>       ;
2850                              <1>       ; INPUT ->
2851                              <1>       ;      ESI = Dot File Name Location
2852                              <1>       ; OUTPUT ->
2853                              <1>       ;      cf = 1 -> error code in AL
2854                              <1>       ;           AL = ERR_INV_FILE_NAME (=26)
2855                              <1>       ;               Invalid file name chars
2856                              <1>       ;      cf = 0 -> valid file name
2857                              <1>       ;
2858                              <1>       ;(EAX, ECX, EDI will be changed)
2859                              <1>
2860                              <1> check_invalid_filename_chars:
2861                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
2862                              <1>       ; 10/07/2010 (FILE.ASM, 'proc_check_invalid_filename_chars')
2863                              <1>       ; 10/02/2010
2864                              <1>       ; Derived from 'proc_check_invalid_filename_chars'
2865                              <1>       ; in the old TRDOS.ASM (09/02/2005).
2866                              <1>       ;
2867                              <1>       ; INPUT ->
2868                              <1>       ;      ESI = ASCIIZ FileName
2869                              <1>       ; OUTPUT ->
2870                              <1>       ;      cf = 1 -> invalid
2871                              <1>       ;      cf = 0 -> valid
2872                              <1>       ;
2873                              <1>       ;(EAX, ECX, EDI will be changed)
2874                              <1>
2875 0000984F 56                  <1>       push   esi
2876                              <1>
2877 00009850 BF[A5410100]        <1>         mov     edi, invalid_fname_chars
2878 00009855 AC                  <1>       lodsb
2879                              <1> check_filename_next_char:
2880 00009856 B914000000          <1>       mov    ecx, sizeInvFnChars
```

```
2881 0000985B BF[A5410100]        <1>       mov    edi, invalid_fname_chars
2882                              <1> loc_scan_invalid_filename_char:
2883 00009860 AE                  <1>       scasb
2884 00009861 741F                <1>       je     short loc_invalid_filename_stc
2885 00009863 E2FB                <1>       loop   loc_scan_invalid_filename_char
2886 00009865 AC                  <1>       lodsb
2887 00009866 3C1F                <1>       cmp    al, 1Fh  ; 20h and above
2888 00009868 77EC                <1>       ja     short check_filename_next_char
2889                              <1>
2890                              <1> check_filename_dot:
2891 0000986A 8B3424              <1>       mov    esi, [esp]
2892                              <1>
2893 0000986D B421                <1>       mov    ah, 21h
2894 0000986F B908000000          <1>       mov    ecx, 8
2895                              <1> loc_check_filename_next_char:
2896 00009874 AC                  <1>       lodsb
2897 00009875 3C2E                <1>       cmp    al, 2Eh
2898 00009877 7511                <1>       jne    short pass_check_fn_dot_check
2899                              <1> loc_check_filename_ext_0:
2900 00009879 AC                  <1>       lodsb
2901 0000987A 38E0                <1>       cmp    al, ah ; 21h
2902 0000987C 7205                <1>       jb     short loc_invalid_filename
2903 0000987E 3C2E                <1>       cmp    al, 2Eh
2904 00009880 7519                <1>       jne    short loc_check_filename_ext_1
2905                              <1>
2906                              <1> loc_invalid_filename_stc:
2907                              <1> loc_check_fn_stc_rtn:
2908 00009882 F9                  <1>       stc
2909                              <1> loc_invalid_filename:
2910                              <1>       ; 10/10/2016 (0Bh -> 26)
2911 00009883 B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME ; (=26)
2912                              <1>       ; Invalid file name chars
2913                              <1> loc_check_fn_rtn:
2914 00009888 5E                  <1>       pop    esi
2915 00009889 C3                  <1>       retn
2916                              <1>
2917                              <1> pass_check_fn_dot_check:
2918 0000988A 38E0                <1>       cmp    al, ah ; 21h
2919 0000988C 7224                <1>       jb     short loc_check_fn_clc_rtn
2920 0000988E E2E4                <1>       loop   loc_check_filename_next_char
2921 00009890 AC                  <1>       lodsb
2922 00009891 38E0                <1>       cmp    al, ah ; 21h
2923 00009893 721D                <1>       jb     short loc_check_fn_clc_rtn
2924 00009895 3C2E                <1>       cmp    al, 2Eh
2925 00009897 75E9                <1>       jne    short loc_check_fn_stc_rtn
2926 00009899 EBDE                <1>       jmp    short loc_check_filename_ext_0
2927                              <1>
2928                              <1> loc_check_filename_ext_1:
2929 0000989B AC                  <1>       lodsb
2930 0000989C 38E0                <1>       cmp    al, ah ; 21h
2931 0000989E 7212                <1>       jb     short loc_check_fn_clc_rtn
2932 000098A0 3C2E                <1>       cmp    al, 2Eh
2933 000098A2 74DE                <1>       je     short loc_check_fn_stc_rtn
2934 000098A4 AC                  <1>       lodsb
2935 000098A5 38E0                <1>       cmp    al, ah ; 21h
2936 000098A7 7209                <1>       jb     short loc_check_fn_clc_rtn
2937 000098A9 3C2E                <1>       cmp    al, 2Eh
2938 000098AB 74D5                <1>       je     short loc_check_fn_stc_rtn
2939 000098AD AC                  <1>       lodsb
2940 000098AE 38E0                <1>       cmp    al, ah ; 21h
2941 000098B0 73D0                <1>       jnb    short loc_check_fn_stc_rtn
2942                              <1>
2943                              <1> loc_check_fn_clc_rtn:
2944 000098B2 5E                  <1>       pop    esi
2945 000098B3 F8                  <1>       clc
2946 000098B4 C3                  <1>       retn
2947                              <1>
2948                              <1> loc_print_deleted_message:
2949 000098B5 BE[91450100]        <1>       mov    esi, Msg_Deleted
2950 000098BA E8A6DCFFFF          <1>       call   print_msg
2951                              <1>
2952                              <1>       ;clc
2953                              <1>
2954                              <1> loc_file_rw_restore_retn:
2955                              <1>       ; 15/02/2016 (TRDOS 386 = TRDOS v2.0)
2956                              <1>       ; 28/02/2010 (CMD_INTR.ASM)
2957                              <1> loc_file_rw_cmd_failed:
2958 000098BF 9C                  <1>       pushf
2959 000098C0 E84FF7FFFF          <1>       call   restore_cdir_after_cmd_fail
2960 000098C5 9D                  <1>       popf
2961 000098C6 720D                <1>       jc     short loc_file_rw_check_write_fault
2962 000098C8 C3                  <1>       retn
2963                              <1>
2964                              <1> loc_permission_denied:
2965                              <1>       ; 27/02/2016
2966 000098C9 BE[9E450100]        <1>       mov    esi, Msg_Permission_Denied
2967 000098CE E892DCFFFF          <1>       call   print_msg
2968 000098D3 EBEA                <1>       jmp    short loc_file_rw_restore_retn
2969                              <1>
2970                              <1> loc_file_rw_check_write_fault:
2971                              <1>       ;cmp   al, 1Dh ; Write Fault
2972 000098D5 3C12                <1>         cmp  al, 18 ; 05/11/2016
2973 000098D7 0F85D4F6FFFF        <1>       jne     loc_run_cmd_failed_cmp_al
2974 000098DD BE[86430100]        <1>       mov    esi, Msg_Not_Ready_Write_Err
2975                              <1>       ;call  print_msg
2976                              <1>       ;retn
2977 000098E2 E97EDCFFFF          <1>       jmp    print_msg
2978                              <1>
2979                              <1> make_directory:
2980                              <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
2981                              <1>       ; 12/03/2011 (CMD_INTR.ASM, 'cmp_cmd_mkdir')
2982                              <1>       ; 14/08/2010
2983                              <1>       ; 10/07/2010
2984                              <1>       ; 29/11/2009
2985                              <1>       ;
```

```
                            <1> get_mkdir_fchar:
                            <1>       ; esi = directory name
000098E7 803E20             <1>       cmp   byte [esi], 20h
000098EA 7701               <1>         ja  short loc_mkdir_parse_path_name
                            <1>
                            <1> loc_mkdir_nodirname_retn:
000098EC C3                 <1>       retn
                            <1>
                            <1> loc_mkdir_parse_path_name:
000098ED BF[7E930100]       <1>       mov   edi, FindFile_Drv
000098F2 E86B1D0000         <1>         call    parse_path_name
000098F7 0F8284F6FFFF       <1>       jc    loc_cmd_failed
                            <1>
                            <1> loc_mkdir_check_dirname_exists:
000098FD BE[C0930100]       <1>       mov   esi, FindFile_Name
00009902 803E20             <1>       cmp   byte [esi], 20h
00009905 0F8676F6FFFF       <1>       jna   loc_cmd_failed
0000990B 8935[3C940100]     <1>       mov   [DelFile_FNPointer], esi
00009911 E839FFFFFF         <1>       call  check_filename
00009916 7259               <1>       jc    short loc_mkdir_invalid_dir_name_chars
                            <1>
                            <1> loc_mkdir_drv:
00009918 8A35[DE8A0100]     <1>       mov   dh, [Current_Drv]
0000991E 8835[3A920100]     <1>       mov   [RUN_CDRV], dh
                            <1>
00009924 8A15[7E930100]     <1>       mov   dl, [FindFile_Drv]
0000992A 38F2               <1>       cmp   dl, dh
0000992C 7407               <1>       je    short loc_mkdir_change_directory
                            <1>
0000992E E8BBE7FFFF         <1>       call  change_current_drive
00009933 728A               <1>       jc    loc_file_rw_cmd_failed
                            <1>
                            <1> loc_mkdir_change_directory:
00009935 803D[7F930100]20   <1>       cmp   byte [FindFile_Directory], 20h
0000993C 7614               <1>       jna   short loc_mkdir_find_directory
                            <1>
0000993E FE05[A6400100]     <1>       inc   byte [Restore_CDIR]
00009944 BE[7F930100]       <1>       mov   esi, FindFile_Directory
00009949 30E4               <1>       xor   ah, ah ; CD_COMMAND sign -> 0
0000994B E8FC160000         <1>       call  change_current_directory
00009950 722E               <1>       jc    short loc_mkdir_check_error_code
                            <1>
                            <1> ;loc_mkdir_change_prompt_dir_string:
                            <1>       ;call  change_prompt_dir_string
                            <1>
                            <1> loc_mkdir_find_directory:
                            <1>       ;mov  esi, FindFile_Name
00009952 8B35[3C940100]     <1>       mov   esi, [DelFile_FNPointer]
                            <1>       ;xor  eax, eax
00009958 6631C0             <1>       xor   ax, ax ; any name (dir, file, volume)
0000995B E831FBFFFF         <1>       call  find_first_file
00009960 721E               <1>       jc    short loc_mkdir_check_error_code
                            <1>
                            <1> loc_mkdir_directory_found:
00009962 BE[E9440100]       <1>       mov   esi, Msg_Name_Exists
00009967 E8F9DBFFFF         <1>       call  print_msg
                            <1>
0000996C E94EFFFFFF         <1>         jmp     loc_file_rw_restore_retn
                            <1>
                            <1> loc_mkdir_invalid_dir_name_chars:
00009971 BE[BC440100]       <1>       mov   esi, Msg_invalid_name_chars
00009976 E8EADBFFFF         <1>       call  print_msg
                            <1>
0000997B E93FFFFFFF         <1>         jmp     loc_file_rw_restore_retn
                            <1>
                            <1> loc_mkdir_check_error_code:
00009980 3C02               <1>       cmp   al, 2
                            <1>       ;je   short loc_mkdir_directory_not_found
00009982 7406               <1>       je    short loc_mkdir_ask_for_yes_no
00009984 F9                 <1>       stc
00009985 E935FFFFFF         <1>         jmp     loc_file_rw_cmd_failed
                            <1>
                            <1> loc_mkdir_directory_not_found:
                            <1> loc_mkdir_ask_for_yes_no:
0000998A BE[0A450100]       <1>       mov   esi, Msg_DoYouWantMkdir
0000998F E8D1DBFFFF         <1>       call  print_msg
00009994 8B35[3C940100]     <1>       mov   esi, [DelFile_FNPointer]
0000999A E8C6DBFFFF         <1>       call  print_msg
0000999F BE[29450100]       <1>       mov   esi, Msg_YesNo
000099A4 E8BCDBFFFF         <1>       call  print_msg
                            <1>
000099A9 C605[33450100]20   <1>       mov   byte [Y_N_nextline], 20h
                            <1>
                            <1> loc_mkdir_ask_again:
000099B0 30E4               <1>       xor   ah, ah
000099B2 E83675FFFF         <1>       call  int16h
000099B7 3C1B               <1>       cmp   al, 1Bh
                            <1>       ;je   short loc_do_not_make_directory
000099B9 7439               <1>       je    short loc_mkdir_y_n_escape
000099BB 24DF               <1>       and   al, 0DFh ; y -> Y, n -> N
000099BD 3C59               <1>       cmp   al, 'Y' ; 'yes'
000099BF 7404               <1>       je    short loc_mkdir_yes_make_directory
000099C1 3C4E               <1>       cmp   al, 'N' ; 'no'
000099C3 75EB               <1>       jne   short loc_mkdir_ask_again
                            <1>
                            <1> loc_do_not_make_directory:
                            <1> loc_mkdir_yes_make_directory:
000099C5 E82E000000         <1>       call  y_n_answer ; 29/12/2017
                            <1>       ;cmp  al, 'Y' ; 'yes'
                            <1>       ;cmc
                            <1>         ;jnc loc_file_rw_restore_retn
000099CA 3C4E               <1>       cmp   al, 'N' ; 'no'
000099CC 0F84EDFEFFFF       <1>         je  loc_file_rw_restore_retn
                            <1>
                            <1> loc_mkdir_call_make_sub_directory:
```

```
3091 000099D2 8B35[3C940100]      <1>       mov    esi, [DelFile_FNPointer]
3092 000099D8 B110                <1>       mov    cl, 10h ; Directory attributes
3093 000099DA E8821D0000          <1>       call   make_sub_directory
3094                              <1> loc_rename_file_ok: ; 06/03/2016
3095 000099DF 0F82DAFEFFFF        <1>         jc   loc_file_rw_cmd_failed
3096                              <1> move_source_file_to_destination_OK:
3097 000099E5 BE[37450100]        <1>       mov    esi, Msg_OK
3098 000099EA E876DBFFFF          <1>       call   print_msg
3099 000099EF E9CBFEFFFF          <1>       jmp    loc_file_rw_restore_retn
3100                              <1>
3101                              <1> loc_mkdir_y_n_escape:
3102 000099F4 B04E                <1>       mov    al, 'N' ; 'no'
3103 000099F6 EBCD                <1>       jmp    short loc_do_not_make_directory
3104                              <1>
3105                              <1> y_n_answer:
3106                              <1>       ; 29/12/2017
3107 000099F8 A2[33450100]        <1>       mov    [Y_N_nextline], al
3108                              <1>       ;push  ax
3109 000099FD 50                  <1>       push   eax
3110 000099FE BE[33450100]        <1>       mov    esi, Y_N_nextline
3111 00009A03 E85DDBFFFF          <1>       call   print_msg
3112 00009A08 58                  <1>       pop    eax
3113                              <1>       ;pop   ax
3114 00009A09 C3                  <1>       retn
3115                              <1>
3116                              <1> delete_directory:
3117                              <1>       ; 29/12/2017
3118                              <1>       ; 15/10/2016
3119                              <1>       ; 01/03/2016, 06/03/2016
3120                              <1>       ; 27/02/2016, 28/02/2016, 29/02/2016
3121                              <1>       ; 26/02/2016 (TRDOS 386 = TRDOS v2.0)
3122                              <1>       ; 16/10/2010 (CMD_INTR.ASM, 'cmp_cmd_rmdir')
3123                              <1>       ; 05/06/2010
3124                              <1>       ;
3125                              <1> get_fchar:
3126                              <1>       ; esi = directory name
3127 00009A0A 803E20              <1>       cmp    byte [esi], 20h
3128 00009A0D 7701                <1>         ja   short loc_rmdir_parse_path_name
3129                              <1>
3130                              <1> loc_rmdir_nodirname_retn:
3131 00009A0F C3                  <1>       retn
3132                              <1>
3133                              <1> loc_rmdir_parse_path_name:
3134 00009A10 BF[7E930100]        <1>       mov    edi, FindFile_Drv
3135 00009A15 E8481C0000          <1>       call   parse_path_name
3136 00009A1A 0F8261F5FFFF        <1>       jc     loc_cmd_failed
3137                              <1>
3138                              <1> loc_rmdir_check_dirname_exists:
3139 00009A20 BE[C0930100]        <1>       mov    esi, FindFile_Name
3140 00009A25 803E20              <1>       cmp    byte [esi], 20h
3141 00009A28 0F8653F5FFFF        <1>       jna    loc_cmd_failed
3142 00009A2E 8935[3C940100]      <1>       mov    [DelFile_FNPointer], esi
3143                              <1>
3144                              <1> loc_rmdir_drv:
3145 00009A34 8A35[DE8A0100]      <1>       mov    dh, [Current_Drv]
3146 00009A3A 8835[3A920100]      <1>       mov    [RUN_CDRV], dh
3147                              <1>
3148 00009A40 8A15[7E930100]      <1>       mov    dl, [FindFile_Drv]
3149 00009A46 38F2                <1>       cmp    dl, dh
3150 00009A48 740B                <1>       je     short loc_rmdir_change_directory
3151                              <1>
3152 00009A4A E89FE6FFFF          <1>       call   change_current_drive
3153 00009A4F 0F826AFEFFFF        <1>       jc     loc_file_rw_cmd_failed
3154                              <1>
3155                              <1> loc_rmdir_change_directory:
3156 00009A55 803D[7F930100]20    <1>       cmp    byte [FindFile_Directory], 20h
3157 00009A5C 7614                <1>       jna    short loc_rmdir_find_directory
3158                              <1>
3159 00009A5E FE05[A6400100]      <1>       inc    byte [Restore_CDIR]
3160 00009A64 BE[7F930100]        <1>       mov    esi, FindFile_Directory
3161 00009A69 30E4                <1>       xor    ah, ah ; CD_COMMAND sign -> 0
3162 00009A6B E8DC150000          <1>       call   change_current_directory
3163 00009A70 7211                <1>       jc     short loc_rmdir_check_error_code
3164                              <1>
3165                              <1> ;loc_rmdir_change_prompt_dir_string:
3166                              <1>       ;call  change_prompt_dir_string
3167                              <1>
3168                              <1> loc_rmdir_find_directory:
3169                              <1>       ;mov   esi, FindFile_Name
3170 00009A72 8B35[3C940100]      <1>       mov    esi, [DelFile_FNPointer]
3171 00009A78 66B81008            <1>       mov    ax, 0810h ; Only directories
3172 00009A7C E810FAFFFF          <1>       call   find_first_file
3173 00009A81 730A                <1>       jnc    short loc_rmdir_ambgfn_check
3174                              <1>
3175                              <1> loc_rmdir_check_error_code:
3176 00009A83 3C02                <1>       cmp    al, 2
3177 00009A85 740B                <1>       je     short loc_rmdir_directory_not_found
3178 00009A87 F9                  <1>       stc
3179 00009A88 E932FEFFFF          <1>       jmp    loc_file_rw_cmd_failed
3180                              <1>
3181                              <1> loc_rmdir_ambgfn_check:
3182 00009A8D 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
3183 00009A90 740F                <1>       jz     short loc_rmdir_directory_found
3184                              <1>
3185                              <1> loc_rmdir_directory_not_found:
3186 00009A92 BE[A8430100]        <1>       mov    esi, Msg_Dir_Not_Found
3187 00009A97 E8C9DAFFFF          <1>       call   print_msg
3188                              <1>
3189 00009A9C E91EFEFFFF          <1>       jmp    loc_file_rw_restore_retn
3190                              <1>
3191                              <1> loc_rmdir_directory_found:
3192 00009AA1 80E307              <1>       and    bl, 07h ; Attributes
3193 00009AA4 0F851FFEFFFF        <1>       jnz    loc_permission_denied
3194                              <1>
3195                              <1> loc_rmdir_save_lnel: ; 28/02/2016
```

```
                                <1>         ;mov  bh, [LongName_EntryLength]
00009AAA 883D[46940100]         <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:
00009AB0 57                     <1>       push  edi ; * (29/02/2016)
                                <1>
00009AB1 BE[3D450100]           <1>       mov   esi, Msg_DoYouWantRmDir
00009AB6 E8AADAFFFF             <1>       call  print_msg
00009ABB 8B35[3C940100]         <1>       mov   esi, [DelFile_FNPointer]
00009AC1 E89FDAFFFF             <1>       call  print_msg
00009AC6 BE[29450100]           <1>       mov   esi, Msg_YesNo
00009ACB E895DAFFFF             <1>       call  print_msg
                                <1>
                                <1> loc_rmdir_ask_again:
00009AD0 30E4                   <1>       xor   ah, ah
00009AD2 E81674FFFF             <1>       call  int16h
00009AD7 3C1B                   <1>       cmp   al, 1Bh
                                <1>       ;je   short loc_do_not_delete_directory
00009AD9 7433                   <1>         je      loc_rmdir_y_n_escape ; 06/03/2016
00009ADB 24DF                   <1>       and   al, 0DFh
00009ADD A2[33450100]           <1>       mov   [Y_N_nextline], al
00009AE2 3C59                   <1>       cmp   al, 'Y'
00009AE4 7404                   <1>       je    short loc_rmdir_yes_delete_directory
00009AE6 3C4E                   <1>       cmp   al, 'N'
00009AE8 75E6                   <1>       jne   short loc_rmdir_ask_again
                                <1>
                                <1> loc_do_not_delete_directory:
                                <1> loc_rmdir_yes_delete_directory:
00009AEA E809FFFFFF             <1>       call  y_n_answer ; 29/12/2017
00009AEF 5F                     <1>       pop   edi ; * (29/02/2016)
                                <1>       ;cmp  al, 'Y' ; 'yes'
                                <1>       ;cmc
                                <1>         ;jnc loc_file_rw_restore_retn
00009AF0 3C4E                   <1>       cmp   al, 'N' ; 'no'
00009AF2 0F84C7FDFFFF           <1>         je  loc_file_rw_restore_retn
                                <1>
                                <1>       ; 29/12/2017
00009AF8 E869000000             <1>       call  delete_sub_directory
00009AFD 7213                   <1>       jc    short loc_rmdir_cmd_failed
                                <1>
                                <1> loc_rmdir_ok:
00009AFF BE[37450100]           <1>       mov   esi, Msg_OK
00009B04 E85CDAFFFF             <1>       call  print_msg
00009B09 E9B1FDFFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_rmdir_y_n_escape:
00009B0E B04E                   <1>       mov   al, 'N' ; 'no'
00009B10 EBD8                   <1>         jmp     loc_do_not_delete_directory
                                <1>
                                <1> loc_rmdir_cmd_failed:
                                <1>       ; 29/12/2017
00009B12 09C0                   <1>       or    eax, eax ; EAX = 0 -> Directory not empty!
00009B14 7426                   <1>       jz    short loc_rmdir_directory_not_empty
                                <1>
                                <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
                                <1>
00009B16 833D[FA910100]01       <1>       cmp   dword [FAT_ClusterCounter], 1
00009B1D 0F829CFDFFFF           <1>       jb    loc_file_rw_cmd_failed
00009B23 F9                     <1>       stc
                                <1> loc_rmdir_cmd_return:
                                <1>       ; 01/03/2016
00009B24 9C                     <1>       pushf
                                <1>       ; ESI = Logical DOS Drive Description Table address
00009B25 66BB00FF               <1>       mov   bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
                                <1>                 ; BL = 0 -> Recalculate free cluster count
00009B29 50                     <1>       push  eax
00009B2A E8C3380000             <1>       call  calculate_fat_freespace
00009B2F 58                     <1>       pop   eax
00009B30 9D                     <1>       popf
00009B31 0F8288FDFFFF           <1>       jc    loc_file_rw_cmd_failed
00009B37 E983FDFFFF             <1>       jmp   loc_file_rw_restore_retn
                                <1>
                                <1> loc_rmdir_directory_not_empty:
00009B3C BE[5E450100]           <1>       mov   esi, Msg_Dir_Not_Empty
00009B41 E81FDAFFFF             <1>       call  print_msg
                                <1>       ; 01/03/2016
00009B46 A1[FA910100]           <1>       mov   eax, [FAT_ClusterCounter]
00009B4B 09C0                   <1>       or    eax, eax ; 0 ?
00009B4D 0F846CFDFFFF           <1>       jz    loc_file_rw_restore_retn
                                <1>       ; ESI = Logical DOS Drive Description Table address
00009B53 66BB01FF               <1>       mov   bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
                                <1>                 ; BL = 1 -> add free clusters
00009B57 E896380000             <1>       call  calculate_fat_freespace
00009B5C 09C9                   <1>       or    ecx, ecx
00009B5E 0F845BFDFFFF           <1>         jz      loc_file_rw_restore_retn ; ecx = 0 -> OK
                                <1>       ; ecx > 0 -> Error (Recalculation is needed)
00009B64 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:
```

```
3301 00009B66 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
3302 00009B6A C1E010              <1>         shl  eax, 16
3303 00009B6D 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
3304                              <1>
3305                              <1>       ;mov   [DelFile_FCluster], eax
3306                              <1>
3307                              <1>       ;;mov  bx, [DirBuff_EntryCounter]
3308                              <1>       ;mov   bx, [FindFile_DirEntryNumber] ; 27/02/2016
3309                              <1>       ;mov   [DelFile_EntryCounter], bx
3310                              <1>
3311 00009B71 29DB                <1>       sub    ebx, ebx
3312                              <1>       ; 29/12/2017
3313 00009B73 891D[FA910100]      <1>       mov    [FAT_ClusterCounter], ebx ; 0 ; Reset
3314                              <1>
3315 00009B79 8A3D[7E930100]      <1>       mov    bh, [FindFile_Drv]
3316 00009B7F BE00010900          <1>       mov    esi, Logical_DOSDisks
3317 00009B84 01DE                <1>       add    esi, ebx
3318                              <1>
3319 00009B86 66817F0CA101        <1>       cmp    word [edi+DirEntry_NTRes], 01A1h
3320 00009B8C 745A                <1>       je     short loc_rmdir_check_fs_directory
3321                              <1>
3322                              <1>       ;cmp   byte [esi+LD_FATType], 1
3323                              <1>       ;jb    short loc_rmdir_get__last_cluster_0
3324                              <1>
3325                              <1>       ; 29/12/2017
3326 00009B8E 83F802              <1>       cmp    eax, 2
3327 00009B91 7306                <1>       jnb    short loc_rmdir_get_last_cluster_1
3328                              <1>       ; eax < 2
3329                              <1> loc_rmdir_get_last_cluster_0:
3330                              <1>       ;mov   eax, ERR_INV_FORMAT ; invalid format!
3331 00009B93 B813000000          <1>       mov    eax, ERR_NOT_DIR ; not a valid directory!
3332                              <1>       ;stc
3333 00009B98 C3                  <1>       retn
3334                              <1>
3335                              <1> loc_rmdir_get_last_cluster_1:
3336 00009B99 807E0303            <1>       cmp    byte [esi+LD_FATType], 3 ; FAT32
3337 00009B9D 750C                <1>       jne    short loc_rmdir_get_last_cluster_2
3338                              <1>
3339                              <1>       ; is it root directory ?
3340 00009B9F 3B4632              <1>       cmp    eax, [esi+LD_BPB+BPB_RootClus]
3341 00009BA2 7507                <1>       jne    short loc_rmdir_get_last_cluster_2
3342                              <1>
3343                              <1>       ; root directory can not be deleted !!
3344                              <1> loc_rmdir_permission_denied:
3345 00009BA4 B80B000000          <1>       mov    eax, ERR_PERM_DENIED ; permission denied!
3346 00009BA9 F9                  <1>       stc
3347 00009BAA C3                  <1>       retn
3348                              <1>
3349                              <1> loc_rmdir_get_last_cluster_2:
3350                              <1>       ; 29/12/2017
3351 00009BAB A3[40940100]        <1>       mov    [DelFile_FCluster], eax
3352                              <1>
3353                              <1>       ;mov   dx, [DirBuff_EntryCounter]
3354 00009BB0 668B15[F8930100]    <1>       mov    dx, [FindFile_DirEntryNumber] ; 27/02/2016
3355 00009BB7 668915[44940100]    <1>       mov    [DelFile_EntryCounter], dx
3356                              <1>
3357 00009BBE 8B15[09920100]      <1>       mov    edx, [DirBuff_Cluster]
3358 00009BC4 8915[70940100]      <1>       mov    [RmDir_ParentDirCluster], edx
3359                              <1>
3360 00009BCA 893D[6C940100]      <1>       mov    [RmDir_DirEntryOffset], edi
3361                              <1>
3362                              <1>       ; 01/03/2016
3363                              <1>       ;mov   dword [FAT_ClusterCounter], 0 ; Reset
3364                              <1>
3365                              <1> loc_rmdir_get_last_cluster_3:
3366 00009BD0 E89C390000          <1>       call   get_last_cluster
3367                              <1>         ;jc  loc_rmdir_cmd_failed
3368 00009BD5 721E                <1>       jc     short loc_delete_sub_dir_retn ; 29/12/2017
3369                              <1>
3370 00009BD7 3B05[40940100]      <1>       cmp    eax, [DelFile_FCluster]
3371 00009BDD 7517                <1>       jne    short loc_rmdir_multi_dir_clusters
3372                              <1>
3373 00009BDF C605[6B940100]00    <1>       mov    byte [RmDir_MultiClusters], 0
3374 00009BE6 EB15                <1>       jmp    short pass_rmdir_multi_dir_clusters
3375                              <1>
3376                              <1> loc_rmdir_check_fs_directory:
3377                              <1>       ; 29/12/2017
3378 00009BE8 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
3379 00009BEC 75B6                <1>       jne    short loc_rmdir_permission_denied
3380                              <1>
3381                              <1> loc_rmdir_delete_fs_directory:
3382 00009BEE E876130000          <1>       call   delete_fs_directory
3383                              <1>       ;jnc   loc_print_deleted_message
3384 00009BF3 7300                <1>       jnc    short loc_delete_sub_dir_retn ; 29/12/2017
3385                              <1>
3386                              <1>       ; EAX=0 -> Directory not empty !
3387                              <1>       ; EAX>0 -> Disk r/w error or another (misc) error
3388                              <1>
3389                              <1>       ;or    eax, eax
3390                              <1>       ;jz    loc_rmdir_directory_not_empty_2
3391                              <1>       ;;stc
3392                              <1>       ;;jmp  loc_file_rw_cmd_failed
3393                              <1>
3394                              <1> loc_delete_sub_dir_retn:
3395 00009BF5 C3                  <1>       retn
3396                              <1>
3397                              <1> loc_rmdir_multi_dir_clusters:
3398 00009BF6 C605[6B940100]01    <1>       mov    byte [RmDir_MultiClusters], 1
3399                              <1>
3400                              <1> pass_rmdir_multi_dir_clusters:
3401 00009BFD A3[74940100]        <1>       mov    [RmDir_DirLastCluster], eax
3402 00009C02 890D[78940100]      <1>       mov    [RmDir_PreviousCluster], ecx
3403                              <1>
3404                              <1> loc_rmdir_load_fat_sub_directory:
3405 00009C08 E84F330000          <1>       call   load_FAT_sub_directory
```

```
3406                                    <1>       ;jc    loc_rmdir_cmd_failed
3407 00009C0D 72E6                      <1>       jc     short loc_delete_sub_dir_retn
3408                                    <1>
3409                                    <1> loc_rmdir_find_last_dir_entry:
3410 00009C0F 56                        <1>       push   esi
3411 00009C10 BE[62930100]              <1>       mov    esi, Dir_File_Name
3412 00009C15 C6062A                    <1>       mov    byte [esi], '*'
3413 00009C18 C646082A                  <1>       mov    byte [esi+8], '*'
3414 00009C1C 31DB                      <1>       xor    ebx, ebx ; Entry offset  = 0
3415                                    <1> loc_rmdir_find_last_dir_entry_next:
3416 00009C1E 66B80008                  <1>       mov    ax, 0800h ; Except volume/long names
3417 00009C22 6631C9                    <1>       xor    cx, cx ; 0 = Find a valid file or dir name
3418 00009C25 E879170000                <1>       call   find_directory_entry
3419 00009C2A 7225                      <1>       jc     short loc_rmdir_empty_dir_cluster
3420 00009C2C 83FB01                    <1>       cmp    ebx, 1
3421 00009C2F 771B                      <1>       ja     short loc_rmdir_directory_not_empty_1
3422                                    <1> loc_rmdir_dot_entry_check:
3423 00009C31 80FD2E                    <1>       cmp    ch, '.' ; The first char of the dir entry
3424 00009C34 7516                      <1>       jne    short loc_rmdir_directory_not_empty_1
3425 00009C36 08DB                      <1>       or     bl, bl
3426 00009C38 7506                      <1>       jnz    short loc_rmdir_dotdot_entry_check
3427 00009C3A 807F0120                  <1>       cmp    byte [edi+1], 20h
3428 00009C3E EB06                      <1>       jmp    short pass_rmdir_dot_entry_check
3429                                    <1>
3430                                    <1> loc_rmdir_dotdot_entry_check:
3431 00009C40 66817F012E20              <1>       cmp    word [edi+1], '. '
3432                                    <1> pass_rmdir_dot_entry_check:
3433 00009C46 7504                      <1>       jne    short loc_rmdir_directory_not_empty_1
3434 00009C48 FEC3                      <1>       inc    bl
3435 00009C4A EBD2                      <1>       jmp    short loc_rmdir_find_last_dir_entry_next
3436                                    <1>
3437                                    <1> loc_rmdir_directory_not_empty_1:
3438 00009C4C 58                        <1>       pop    eax ; pushed esi
3439 00009C4D 31C0                      <1>       xor    eax, eax ; 0
3440                                    <1> loc_rmdir_directory_not_empty_2:
3441                                    <1> loc_delete_sub_dir_stc_retn:
3442 00009C4F F9                        <1>       stc
3443 00009C50 C3                        <1>       retn
3444                                    <1>
3445                                    <1> loc_rmdir_empty_dir_cluster:
3446 00009C51 5E                        <1>       pop    esi
3447                                    <1>
3448                                    <1> loc_rmdir_set_prev_cluster_dir_last_cluster:
3449 00009C52 803D[6B940100]00          <1>       cmp    byte [RmDir_MultiClusters], 0
3450 00009C59 7613                      <1>       jna    short loc_rmdir_unlink_dir_last_cluster
3451                                    <1>
3452 00009C5B A1[78940100]              <1>       mov    eax, [RmDir_PreviousCluster]
3453                                    <1>       ;xor   ecx, ecx
3454 00009C60 49                        <1>       dec    ecx ; FFFFFFFFh
3455 00009C61 E83A340000                <1>       call   update_cluster
3456 00009C66 7306                      <1>       jnc    short loc_rmdir_unlink_dir_last_cluster
3457                                    <1>
3458                                    <1>       ; 01/03/2016
3459                                    <1>       ;cmp   eax, 1  ; eax = 0 -> end of cluster chain
3460                                    <1>       ;cmc
3461                                    <1>       ;jc    short loc_rmdir_cmd_failed
3462                                    <1>       ;jmp   short loc_rmdir_save_fat_buffer
3463                                    <1>       ; 29/12/2017
3464 00009C68 21C0                      <1>       and    eax, eax
3465 00009C6A 75E3                      <1>       jnz    short loc_delete_sub_dir_stc_retn
3466 00009C6C EB12                      <1>       jmp    short loc_rmdir_save_fat_buffer
3467                                    <1>
3468                                    <1> loc_rmdir_unlink_dir_last_cluster:
3469 00009C6E A1[74940100]              <1>       mov    eax, [RmDir_DirLastCluster]
3470 00009C73 31C9                      <1>       xor    ecx, ecx ; 0
3471 00009C75 E826340000                <1>       call   update_cluster
3472 00009C7A 7327                      <1>       jnc    short loc_rmdir_unlink_stc_retn_0Bh
3473                                    <1>       ; Because of it is the last cluster
3474                                    <1>       ; 'update_cluster' must return with eocc error
3475 00009C7C 09C0                      <1>       or     eax, eax
3476                                    <1>       ;jz    short loc_rmdir_save_fat_buffer ; eocc
3477                                    <1>       ;stc
3478                                    <1>         ;jmp     short loc_rmdir_cmd_failed
3479                                    <1>       ; 29/12/2017
3480 00009C7E 75CF                      <1>       jnz    short loc_delete_sub_dir_stc_retn
3481                                    <1>
3482                                    <1> loc_rmdir_save_fat_buffer:
3483 00009C80 803D[F2910100]02          <1>       cmp    byte [FAT_BuffValidData], 2
3484 00009C87 7528                      <1>       jne    short loc_rmdir_calculate_FAT_freespace
3485 00009C89 E8CF360000                <1>       call   save_fat_buffer
3486                                    <1>       ;jc    short loc_rmdir_cmd_failed
3487                                    <1>       ; 29/12/2017
3488 00009C8E 7219                      <1>       jc     short loc_rmdir_unlink_error_retn
3489                                    <1>
3490                                    <1>       ; 01/03/2016
3491 00009C90 803D[6B940100]00          <1>       cmp    byte [RmDir_MultiClusters], 0
3492 00009C97 7618                      <1>       jna    short loc_rmdir_calculate_FAT_freespace
3493                                    <1>
3494 00009C99 A1[40940100]              <1>       mov    eax, [DelFile_FCluster]
3495 00009C9E E92DFFFFFF                <1>         jmp     loc_rmdir_get_last_cluster_3
3496                                    <1>
3497                                    <1> loc_rmdir_unlink_stc_retn_0Bh:
3498                                    <1>       ; 15/10/2016 (0Bh -> 28)
3499 00009CA3 B81C000000                <1>       mov    eax, ERR_INV_FORMAT ; 28 = Invalid format
3500                                    <1> loc_rmdir_unlink_stc_retn:
3501 00009CA8 F9                        <1>       stc
3502                                    <1> loc_rmdir_unlink_error_retn:
3503 00009CA9 C3                        <1>       retn
3504                                    <1>
3505                                    <1> loc_rmdir_delete_short_name_invalid_data:
3506 00009CAA B81D000000                <1>       mov    eax, 29 ; Invalid data (15/10/2016)
3507                                    <1>       ;stc
3508                                    <1>         ;jmp loc_rmdir_cmd_failed
3509                                    <1>       ; 29/12/2017
3510 00009CAF EBF7                      <1>       jmp    short loc_rmdir_unlink_stc_retn
```

```
                            <1>
                            <1> loc_rmdir_calculate_FAT_freespace:
                            <1>       ;mov   eax, [FAT_ClusterCounter]
                            <1>       ; 29/12/2017
00009CB1 29C0               <1>       sub    eax, eax ; 0
00009CB3 8705[FA910100]     <1>       xchg   eax, [FAT_ClusterCounter]
                            <1>       ;
00009CB9 66BB01FF           <1>       mov    bx, 0FF01h
                            <1>       ; BL = 1 -> Add EAX to free space count
                            <1>       ; BH = FFh ->
                            <1>       ; ESI = Logical DOS Drive Description Table address
00009CBD E830370000          <1>       call   calculate_fat_freespace
                            <1>
00009CC2 21C9               <1>       and    ecx, ecx ; ecx = 0 -> valid free sector count
00009CC4 7409               <1>       jz     short loc_rmdir_delete_short_name_continue
                            <1>
                            <1> loc_rmdir_recalculate_FAT_freespace:
00009CC6 66BB00FF           <1>         mov     bx, 0FF00h ; BL = 0 -> Recalculate free space
00009CCA E823370000          <1>       call   calculate_fat_freespace
                            <1>
                            <1> loc_rmdir_delete_short_name_continue:
00009CCF A1[70940100]       <1>       mov    eax, [RmDir_ParentDirCluster]
00009CD4 83F802             <1>       cmp    eax, 2
00009CD7 7309               <1>       jnb    short loc_rmdir_del_short_name_load_sub_dir
00009CD9 E8F3310000         <1>       call   load_FAT_root_directory
                            <1>       ;jc    loc_file_rw_cmd_failed
                            <1>       ; 29/12/2017
00009CDE 72C9               <1>       jc     short loc_rmdir_unlink_error_retn
00009CE0 EB07               <1>       jmp    short loc_rmdir_del_short_name_ld_chk_fclust
                            <1>
                            <1> loc_rmdir_del_short_name_load_sub_dir:
00009CE2 E875320000         <1>       call   load_FAT_sub_directory
                            <1>       ;jc    loc_file_rw_cmd_failed
                            <1>       ; 29/12/2017
00009CE7 72C0               <1>       jc     short loc_rmdir_unlink_error_retn
                            <1>
                            <1> loc_rmdir_del_short_name_ld_chk_fclust:
00009CE9 0FB73D[6C940100]   <1>       movzx  edi, word [RmDir_DirEntryOffset]
00009CF0 81C700000800       <1>       add    edi, Directory_Buffer
                            <1>
00009CF6 668B4714           <1>       mov    ax, [edi+20] ; First Cluster High Word
00009CFA C1E010             <1>       shl    eax, 16
00009CFD 668B471A           <1>       mov    ax, [edi+26] ; First Cluster Low Word
                            <1>         ; Not necessary...
00009D01 3B05[40940100]     <1>       cmp    eax, [DelFile_FCluster]
00009D07 75A1               <1>       jne    short loc_rmdir_delete_short_name_invalid_data
                            <1>       ;
00009D09 C607E5             <1>       mov    byte [edi], 0E5h ; 'Deleted' sign
                            <1>       ; 27/02/2016
                            <1>       ; TRDOS v1 has a bug here! it does not set
                            <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
                            <1>       ; 'save_directory_buffer' would not save the change !
00009D0C C605[04920100]02   <1>       mov    byte [DirBuff_ValidData], 2 ; change sign
                            <1>       ;
00009D13 E8AE1D0000         <1>       call   save_directory_buffer
                            <1>       ;jc    loc_file_rw_cmd_failed
                            <1>       ; 29/12/2017
00009D18 728F               <1>       jc     short loc_rmdir_unlink_error_retn
                            <1>
                            <1> loc_rmdir_del_long_name:
00009D1A 0FB615[46940100]   <1>       movzx  edx, byte [DelFile_LNEL]
00009D21 08D2               <1>       or     dl, dl
00009D23 7410               <1>       jz     short loc_rmdir_update_parent_dir_lmdt
                            <1>
00009D25 0FB705[44940100]   <1>       movzx  eax, word [DelFile_EntryCounter]
00009D2C 29D0               <1>       sub    eax, edx
                            <1>       ; 29/12/2017
00009D2E 7205               <1>       jc     short loc_rmdir_update_parent_dir_lmdt
                            <1>
                            <1>       ; EAX = Directory Entry Number of the long name last entry
00009D30 E8EF1E0000         <1>       call   delete_longname
                            <1>
                            <1> loc_rmdir_update_parent_dir_lmdt:
00009D35 E8271E0000         <1>       call   update_parent_dir_lmdt
                            <1>       ;jc    short loc_file_rw_cmd_failed
                            <1>       ; 29/12/2017
                            <1>       ;jc    short loc_rmdir_unlink_error_retn
                            <1>
                            <1> loc_delete_sub_directory_ok:
                            <1>       ; 29/12/2017
00009D3A 31C0               <1>       xor    eax, eax ;  0 ;  cf = 0
00009D3C C3                 <1>       retn
                            <1>
                            <1>
                            <1> delete_file:
                            <1>       ; 29/02/2016
                            <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 09/08/2010 (CMD_INTR.ASM, 'cmp_cmd_del')
                            <1>       ; 28/02/2010
                            <1>
                            <1> get_delfile_fchar:
                            <1>       ; esi = file name
00009D3D 803E20             <1>       cmp    byte [esi], 20h
00009D40 7701               <1>         ja   short loc_delfile_parse_path_name
                            <1>
                            <1> loc_delfile_nofilename_retn:
00009D42 C3                 <1>       retn
                            <1>
                            <1> loc_delfile_parse_path_name:
00009D43 BF[7E930100]       <1>       mov    edi, FindFile_Drv
00009D48 E815190000         <1>       call   parse_path_name
00009D4D 0F822EF2FFFF       <1>       jc     loc_cmd_failed
                            <1>
                            <1> loc_delfile_check_filename_exists:
00009D53 BE[C0930100]       <1>       mov    esi, FindFile_Name
```

```
3616 00009D58 803E20              <1>       cmp   byte [esi], 20h
3617 00009D5B 0F8620F2FFFF        <1>       jna   loc_cmd_failed
3618 00009D61 8935[3C940100]      <1>       mov   [DelFile_FNPointer], esi
3619                              <1>
3620                              <1> loc_delfile_drv:
3621 00009D67 8A15[7E930100]      <1>       mov   dl, [FindFile_Drv]
3622 00009D6D 8A35[DE8A0100]      <1>       mov   dh, [Current_Drv]
3623 00009D73 8835[3A920100]      <1>       mov   [RUN_CDRV], dh
3624 00009D79 38F2                <1>       cmp   dl, dh
3625 00009D7B 740B                <1>       je    short loc_delfile_change_directory
3626                              <1>
3627 00009D7D E86CE3FFFF          <1>       call  change_current_drive
3628 00009D82 0F8237FBFFFF        <1>       jc    loc_file_rw_cmd_failed
3629                              <1>
3630                              <1> loc_delfile_change_directory:
3631 00009D88 803D[7F930100]20    <1>       cmp   byte [FindFile_Directory], 20h
3632 00009D8F 7618                <1>       jna   short loc_delfile_find
3633                              <1>
3634 00009D91 FE05[A6400100]      <1>       inc   byte [Restore_CDIR]
3635 00009D97 BE[7F930100]        <1>       mov   esi, FindFile_Directory
3636 00009D9C 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
3637 00009D9E E8A9120000          <1>       call  change_current_directory
3638 00009DA3 0F8216FBFFFF        <1>       jc    loc_file_rw_cmd_failed
3639                              <1>
3640                              <1> ;loc_delfile_change_prompt_dir_string:
3641                              <1>       ;call  change_prompt_dir_string
3642                              <1>
3643                              <1> loc_delfile_find:
3644                              <1>       ;mov   esi, FindFile_Name
3645 00009DA9 8B35[3C940100]      <1>       mov   esi, [DelFile_FNPointer]
3646 00009DAF 66B80018            <1>       mov   ax, 1800h ; Except volume label and dirs
3647 00009DB3 E8D9F6FFFF          <1>       call  find_first_file
3648 00009DB8 0F8201FBFFFF        <1>       jc    loc_file_rw_cmd_failed
3649                              <1>
3650                              <1> loc_delfile_ambgfn_check:
3651 00009DBE 6621D2              <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
3652 00009DC1 740B                <1>       jz    short loc_delfile_found
3653                              <1>
3654                              <1> loc_file_not_found:
3655 00009DC3 B802000000          <1>       mov   eax, 2 ; File not found sign
3656 00009DC8 F9                  <1>       stc
3657 00009DC9 E9F1FAFFFF          <1>       jmp   loc_file_rw_cmd_failed
3658                              <1>
3659                              <1> loc_delfile_found:
3660 00009DCE 80E307              <1>       and   bl, 07h ; Attributes
3661 00009DD1 0F85F2FAFFFF        <1>         jnz     loc_permission_denied
3662                              <1>
3663                              <1> ;loc_delfile_found_save_lnel:
3664                              <1> ;     mov   [DelFile_LNEL], bh ; Long name entry length (if > 0)
3665                              <1>
3666                              <1> loc_delfile_ask_for_delete:
3667 00009DD7 57                  <1>       push  edi ; * (29/02/2016)
3668                              <1>
3669 00009DD8 BE[75450100]        <1>       mov   esi, Msg_DoYouWantDelete
3670 00009DDD E883D7FFFF          <1>       call  print_msg
3671 00009DE2 8B35[3C940100]      <1>       mov   esi, [DelFile_FNPointer]
3672 00009DE8 E878D7FFFF          <1>       call  print_msg
3673 00009DED BE[29450100]        <1>       mov   esi, Msg_YesNo
3674 00009DF2 E86ED7FFFF          <1>       call  print_msg
3675                              <1>
3676                              <1> loc_delfile_ask_again:
3677 00009DF7 30E4                <1>       xor   ah, ah
3678 00009DF9 E8EF70FFFF          <1>       call  int16h
3679 00009DFE 3C1B                <1>       cmp   al, 1Bh
3680                              <1>       ;je   short loc_do_not_delete_file
3681 00009E00 7449                <1>       je    short loc_delfile_y_n_escape ; 06/03/2016
3682 00009E02 24DF                <1>       and   al, 0DFh
3683 00009E04 A2[33450100]        <1>       mov   [Y_N_nextline], al
3684 00009E09 3C59                <1>       cmp   al, 'Y'
3685 00009E0B 7404                <1>       je    short loc_yes_delete_file
3686 00009E0D 3C4E                <1>       cmp   al, 'N'
3687 00009E0F 75E6                <1>       jne   short loc_delfile_ask_again
3688                              <1>
3689                              <1> loc_do_not_delete_file:
3690                              <1> loc_yes_delete_file:
3691 00009E11 E8E2FBFFFF          <1>       call  y_n_answer ; 29/12/2017
3692 00009E16 5F                  <1>       pop   edi ; * (29/02/2016)
3693                              <1>       ;cmp  al, 'Y' ; 'yes'
3694                              <1>       ;cmc
3695                              <1>         ;jnc loc_file_rw_restore_retn
3696 00009E17 3C4E                <1>       cmp   al, 'N' ; 'no'
3697 00009E19 0F84A0FAFFFF        <1>         je  loc_file_rw_restore_retn
3698                              <1>
3699                              <1> loc_delete_file:
3700 00009E1F 8A3D[7E930100]      <1>       mov   bh, [FindFile_Drv]
3701                              <1>       ;mov  bl, [DelFile_LNEL]
3702 00009E25 8A1D[CD930100]      <1>       mov   bl, [FindFile_LongNameEntryLength]
3703                              <1>       ;mov  cx, [DirBuff_EntryCounter]
3704 00009E2B 668B0D[F8930100]    <1>       mov   cx, [FindFile_DirEntryNumber]
3705                              <1>       ; (*) EDI = Directory buffer entry offset/address
3706 00009E32 E8D71F0000          <1>       call  remove_file ; (FILE.ASM, 'proc_delete_file')
3707 00009E37 0F8378FAFFFF        <1>       jnc   loc_print_deleted_message
3708                              <1>
3709                              <1>       ;cmp  al, 05h
3710 00009E3D 3C0B                <1>       cmp   al, ERR_PERM_DENIED  ; 29/12/2017 (5 -> 11)
3711 00009E3F 0F8484FAFFFF        <1>       je    loc_permission_denied
3712 00009E45 F9                  <1>       stc
3713 00009E46 E974FAFFFF          <1>       jmp   loc_file_rw_cmd_failed
3714                              <1>
3715                              <1> loc_delfile_y_n_escape:
3716 00009E4B B04E                <1>       mov   al, 'N' ; 'no'
3717 00009E4D EBC2                <1>       jmp   short loc_do_not_delete_file
3718                              <1>
3719                              <1> set_file_attributes:
3720                              <1>       ; 06/03/2016
```

```
                        <1>       ; 04/03/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 10/07/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_attrib')
                        <1>       ; 23/05/2010
                        <1>       ; 17/12/2000 (P2000.ASM)
                        <1>
                        <1>       ; esi = file or directory name
00009E4F 6631C0         <1>       xor    ax, ax
00009E52 66A3[C6450100] <1>       mov    [Attr_Chars], ax
00009E58 A2[94940100]   <1>       mov    [Attributes], al
                        <1>
                        <1> get_attrib_fchar:
                        <1>       ; esi = file name
00009E5D 8A06           <1>       mov    al, [esi]
00009E5F 3C20           <1>       cmp    al, 20h
00009E61 7623           <1>       jna    short loc_attr_file_nofilename_retn
                        <1>
                        <1> loc_scan_attrib_params:
00009E63 3C2D           <1>       cmp    al, '-'
00009E65 0F871C010000   <1>       ja     loc_attr_file_parse_path_name
00009E6B 7408           <1>       je     short loc_attr_space
                        <1>
00009E6D 3C2B           <1>       cmp    al, '+'
00009E6F 0F850CF1FFFF   <1>       jne    loc_cmd_failed
                        <1>
                        <1> loc_attr_space:
00009E75 8A6601         <1>       mov    ah, [esi+1]
00009E78 80FC20         <1>       cmp    ah, 20h
00009E7B 770A           <1>       ja     short pass_attr_space
00009E7D 0F82FEF0FFFF   <1>       jb     loc_cmd_failed
00009E83 46             <1>       inc    esi
00009E84 EBEF           <1>       jmp    short loc_attr_space
                        <1>
                        <1> loc_attr_file_nofilename_retn:
00009E86 C3             <1>       retn
                        <1>
                        <1> pass_attr_space:
00009E87 80E4DF         <1>       and    ah, 0DFh
00009E8A 80FC53         <1>       cmp    ah, 'S'
00009E8D 0F87EEF0FFFF   <1>       ja     loc_cmd_failed
00009E93 7204           <1>       jb     short pass_attr_system
00009E95 B404           <1>       mov    ah, 04h   ; System
00009E97 EB21           <1>       jmp    short pass_attr_archive
                        <1>
                        <1> pass_attr_system:
00009E99 80FC48         <1>       cmp    ah, 'H'
00009E9C 7706           <1>       ja     short pass_attr_hidden
00009E9E 7213           <1>       jb     short pass_attr_read_only
00009EA0 B402           <1>       mov    ah, 02h    ; Hidden
00009EA2 EB16           <1>       jmp    short pass_attr_archive
                        <1>
                        <1> pass_attr_hidden:
00009EA4 80FC52         <1>       cmp    ah, 'R'
00009EA7 0F87D4F0FFFF   <1>       ja     loc_cmd_failed
00009EAD 7204           <1>       jb     short pass_attr_read_only ; Read only
00009EAF B401           <1>       mov    ah, 01h
00009EB1 EB07           <1>       jmp    short pass_attr_archive
                        <1>
                        <1> pass_attr_read_only:
00009EB3 80FC41         <1>       cmp    ah, 'A'
00009EB6 753B           <1>       jne    short loc_chk_attr_enter
00009EB8 B420           <1>       mov    ah, 20h    ; Archive
                        <1>
                        <1> pass_attr_archive:
00009EBA 3C2D           <1>       cmp    al, '-'
00009EBC 7508           <1>       jne    short pass_reducing_attributes
00009EBE 0825[C6450100] <1>       or     [Attr_Chars], ah
00009EC4 EB06           <1>       jmp    short loc_change_attributes_inc
                        <1>
                        <1> pass_reducing_attributes:
00009EC6 0825[C7450100] <1>       or     [Attr_Chars+1], ah
                        <1>
                        <1> loc_change_attributes_inc:
00009ECC 46             <1>       inc    esi
00009ECD 8A6601         <1>       mov    ah, [esi+1]
00009ED0 80FC20         <1>       cmp    ah, 20h
00009ED3 7227           <1>       jb     short pass_change_attr
00009ED5 74F5           <1>       je     short loc_change_attributes_inc
00009ED7 80FC2D         <1>       cmp    ah, '-'
00009EDA 770D           <1>       ja     short loc_chk_next_attr_char1
00009EDC 7405           <1>       je     short loc_chk_next_attr_char0
00009EDE 80FC2B         <1>       cmp    ah, '+'
00009EE1 7506           <1>       jne    short loc_chk_next_attr_char1
                        <1>
                        <1> loc_chk_next_attr_char0:
00009EE3 46             <1>       inc    esi
00009EE4 668B06         <1>       mov    ax, [esi]
00009EE7 EB9E           <1>       jmp    short pass_attr_space
                        <1>
                        <1> loc_chk_next_attr_char1:
00009EE9 803E2D         <1>       cmp    byte [esi], '-'
00009EEC 7799           <1>       ja     short pass_attr_space
00009EEE E988000000     <1>         jmp     loc_attr_file_check_fname_fchar
                        <1>
                        <1> loc_chk_attr_enter:
00009EF3 80FC0D         <1>       cmp    ah, 0Dh
00009EF6 0F8585F0FFFF   <1>       jne    loc_cmd_failed
                        <1>
                        <1> pass_change_attr:
00009EFC A0[C6450100]   <1>       mov    al, [Attr_Chars]
00009F01 F6D0           <1>       not    al
00009F03 2005[94940100] <1>       and    [Attributes], al
00009F09 A0[C7450100]   <1>       mov    al, [Attr_Chars+1]
00009F0E 0805[94940100] <1>       or     [Attributes], al
                        <1>
                        <1> loc_show_attributes:
```

```
3826 00009F14 BE[434D0100]        <1>       mov   esi, nextline
3827 00009F19 E847D6FFFF          <1>       call  print_msg
3828                              <1>
3829                              <1> loc_show_attributes_no_nextline:
3830 00009F1E C705[C6450100]4E4F- <1>       mov   dword [Attr_Chars], 'NORM'
3830 00009F26 524D                <1>
3831 00009F28 66C705[CA450100]41- <1>       mov   word [Attr_Chars+4], 'AL'
3831 00009F30 4C                  <1>
3832 00009F31 BE[C6450100]        <1>       mov   esi, Attr_Chars
3833 00009F36 A0[94940100]        <1>       mov   al, [Attributes]
3834 00009F3B A804                <1>       test  al, 04h
3835 00009F3D 7406                <1>       jz    short pass_put_attr_s
3836 00009F3F 66C7065300          <1>       mov   word [esi], 0053h     ; S
3837 00009F44 46                  <1>       inc   esi
3838                              <1>
3839                              <1> pass_put_attr_s:
3840 00009F45 A802                <1>       test  al, 02h
3841 00009F47 7406                <1>       jz    short pass_put_attr_h
3842 00009F49 66C7064800          <1>       mov   word [esi], 0048h     ; H
3843 00009F4E 46                  <1>       inc   esi
3844                              <1>
3845                              <1> pass_put_attr_h:
3846 00009F4F A801                <1>       test  al, 01h
3847 00009F51 7406                <1>       jz    short pass_put_attr_r
3848 00009F53 66C7065200          <1>       mov   word [esi], 0052h     ; R
3849 00009F58 46                  <1>       inc   esi
3850                              <1>
3851                              <1> pass_put_attr_r:
3852 00009F59 3C20                <1>       cmp   al, 20h
3853 00009F5B 7205                <1>       jb    short pass_put_attr_a
3854 00009F5D 66C7064100          <1>       mov   word [esi], 0041h     ; A
3855                              <1>
3856                              <1> pass_put_attr_a:
3857 00009F62 BE[B9450100]        <1>       mov   esi, Str_Attributes
3858 00009F67 E8F9D5FFFF          <1>       call  print_msg
3859 00009F6C BE[434D0100]        <1>       mov   esi, nextline
3860 00009F71 E8EFD5FFFF          <1>       call  print_msg
3861 00009F76 E944F9FFFF          <1>       jmp   loc_file_rw_restore_retn
3862                              <1>
3863                              <1> loc_attr_file_check_fname_fchar:
3864 00009F7B 46                  <1>       inc   esi
3865 00009F7C 803E20              <1>       cmp   byte [esi], 20h
3866 00009F7F 74FA                <1>       je    short loc_attr_file_check_fname_fchar
3867 00009F81 0F8275FFFFFF        <1>         jb      pass_change_attr
3868                              <1>
3869                              <1> loc_attr_file_parse_path_name:
3870 00009F87 BF[7E930100]        <1>       mov   edi, FindFile_Drv
3871 00009F8C E8D1160000          <1>       call  parse_path_name
3872 00009F91 0F82EAEFFFFF        <1>       jc    loc_cmd_failed
3873                              <1>
3874                              <1> loc_attr_file_check_filename_exists:
3875 00009F97 BE[C0930100]        <1>       mov   esi, FindFile_Name
3876 00009F9C 803E20              <1>       cmp   byte [esi], 20h
3877 00009F9F 0F86DCEFFFFF        <1>       jna   loc_cmd_failed
3878 00009FA5 8935[3C940100]      <1>       mov   [DelFile_FNPointer], esi
3879                              <1>
3880                              <1> loc_attr_file_drv:
3881 00009FAB 8A35[DE8A0100]      <1>       mov   dh, [Current_Drv]
3882 00009FB1 8835[3A920100]      <1>       mov   [RUN_CDRV], dh
3883                              <1>
3884 00009FB7 8A15[7E930100]      <1>       mov   dl, [FindFile_Drv]
3885 00009FBD 38F2                <1>       cmp   dl, dh
3886 00009FBF 740B                <1>       je    short loc_attr_file_change_directory
3887                              <1>
3888 00009FC1 E828E1FFFF          <1>       call  change_current_drive
3889 00009FC6 0F82F3F8FFFF        <1>       jc    loc_file_rw_cmd_failed
3890                              <1>
3891                              <1> loc_attr_file_change_directory:
3892 00009FCC 803D[7F930100]20    <1>         cmp     byte [FindFile_Directory], 20h
3893 00009FD3 7618                <1>       jna   short loc_attr_file_find
3894                              <1>
3895 00009FD5 FE05[A6400100]      <1>       inc   byte [Restore_CDIR]
3896                              <1>
3897 00009FDB BE[7F930100]        <1>       mov   esi, FindFile_Directory
3898 00009FE0 30E4                <1>       xor   ah, ah ; CD_COMMAND sign -> 0
3899 00009FE2 E865100000          <1>       call  change_current_directory
3900 00009FE7 0F82D2F8FFFF        <1>       jc    loc_file_rw_cmd_failed
3901                              <1>
3902                              <1> ;loc_attr_file_change_prompt_dir_string:
3903                              <1>       ;call  change_prompt_dir_string
3904                              <1>
3905                              <1> loc_attr_file_find:
3906                              <1>       ;mov   esi, FindFile_Name
3907 00009FED 8B35[3C940100]      <1>       mov   esi, [DelFile_FNPointer]
3908 00009FF3 66B80008            <1>       mov   ax, 0800h ; Except volume labels
3909 00009FF7 E895F4FFFF          <1>       call  find_first_file
3910 00009FFC 0F82BDF8FFFF        <1>       jc    loc_file_rw_cmd_failed
3911                              <1>
3912                              <1> loc_attr_file_ambgfn_check:
3913 0000A002 6609D2              <1>       or    dx, dx ; Ambiguous filename chars used sign (DX>0)
3914                              <1>       ;     (Note: It was BX in TRDOS v1)
3915                              <1>       ;jz   short loc_attr_file_found
3916 0000A005 0F85B8FDFFFF        <1>         jnz     loc_file_not_found ; 06/03/2016
3917                              <1>
3918                              <1>       ;mov  eax, 2 ; File not found sign
3919                              <1>       ;stc
3920                              <1>       ;jmp  loc_file_rw_cmd_failed
3921                              <1>
3922                              <1> loc_attr_file_found:
3923                              <1>       ; EDI = Directory buffer entry offset/address
3924                              <1>       ; BL = File (or Directory) Attributes
3925                              <1>       ;     (Note: It was 'CL' in TRDOS v1)
3926                              <1>       ; mov  bl, [EDI+0Bh]
3927                              <1>
3928 0000A00B 66833D[C6450100]00  <1>       cmp   word [Attr_Chars], 0
```

```
3929 0000A013 770B                <1>       ja     short loc_attr_file_change_attributes
3930 0000A015 881D[94940100]      <1>       mov    [Attributes], bl
3931 0000A01B E9F4FEFFFF          <1>       jmp    loc_show_attributes
3932                              <1>
3933                              <1> loc_attr_file_change_attributes:
3934 0000A020 A0[C6450100]        <1>       mov    al, [Attr_Chars]
3935 0000A025 F6D0                <1>       not    al
3936 0000A027 20C3                <1>       and    bl, al
3937 0000A029 A0[C7450100]        <1>       mov    al, [Attr_Chars+1]
3938 0000A02E 08C3                <1>       or     bl, al
3939                              <1>
3940 0000A030 66817F0CA101        <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
3941 0000A036 741D                <1>       je     short loc_attr_file_fs_check
3942                              <1>
3943 0000A038 881D[94940100]      <1>       mov    [Attributes], bl
3944 0000A03E 885F0B              <1>       mov    [edi+0Bh], bl    ; Attributes (New!)
3945                              <1>
3946                              <1>       ; 04/03/2016
3947                              <1>       ; TRDOS v1 has a bug here! it does not set
3948                              <1>       ; 'DirBuff_ValidData' to 2; as result of this bug,
3949                              <1>       ; 'save_directory_buffer' would not save the new attributes !
3950                              <1>
3951 0000A041 C605[04920100]02    <1>       mov    byte [DirBuff_ValidData], 2
3952                              <1>
3953 0000A048 E8791A0000          <1>       call   save_directory_buffer
3954 0000A04D 0F826CF8FFFF        <1>       jc     loc_file_rw_cmd_failed
3955                              <1>
3956 0000A053 EB33                <1>       jmp    short loc_print_attr_changed_message
3957                              <1>
3958                              <1> loc_attr_file_fs_check:
3959 0000A055 29C0                <1>       sub    eax, eax
3960 0000A057 8A25[02920100]      <1>         mov     ah, [DirBuff_DRV]
3961 0000A05D BE00010900          <1>       mov    esi, Logical_DOSDisks
3962 0000A062 01C6                <1>         add     esi, eax
3963 0000A064 807E04A1            <1>         cmp     byte [esi+LD_FSType], 0A1h
3964 0000A068 7309                <1>       jnc    short loc_attr_file_change_fs_file_attributes
3965                              <1>       ; 29/12/2017 (0Dh -> 29)
3966 0000A06A 66B81D00            <1>       mov    ax, 29 ; Invalid Data
3967 0000A06E E94CF8FFFF          <1>       jmp    loc_file_rw_cmd_failed
3968                              <1>
3969                              <1> loc_attr_file_change_fs_file_attributes:
3970                              <1>       ; BL = New MS-DOS File Attributes
3971 0000A073 88D8                <1>       mov    al, bl ; File/Directory Attributes
3972 0000A075 30E4                <1>       xor    ah, ah ; Attributes in MS-DOS format sign
3973 0000A077 E873050000          <1>       call   change_fs_file_attributes
3974 0000A07C 0F823DF8FFFF        <1>       jc     loc_file_rw_cmd_failed
3975                              <1>
3976 0000A082 881D[94940100]      <1>       mov    [Attributes], bl
3977                              <1>
3978                              <1> loc_print_attr_changed_message:
3979 0000A088 BE[B4450100]        <1>       mov    esi, Msg_New
3980 0000A08D E8D3D4FFFF          <1>       call   print_msg
3981 0000A092 E987FEFFFF          <1>       jmp    loc_show_attributes_no_nextline
3982                              <1>
3983                              <1> rename_file:
3984                              <1>       ; 13/11/2017
3985                              <1>       ; 06/11/2016
3986                              <1>       ; 05/11/2016
3987                              <1>       ; 16/10/2016
3988                              <1>       ; 08/03/2016
3989                              <1>       ; 06/03/2016 (TRDOS 386 = TRDOS v2.0)
3990                              <1>       ; 20/11/2010 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_rename')
3991                              <1>       ; 16/11/2010
3992                              <1>
3993                              <1> get_rename_source_fchar:
3994                              <1>       ; esi = file name
3995 0000A097 803E20              <1>       cmp    byte [esi], 20h
3996 0000A09A 7614                <1>         jna  short loc_rename_nofilename_retn
3997                              <1>
3998 0000A09C 8935[BC940100]      <1>       mov    [SourceFilePath], esi
3999                              <1>
4000                              <1> rename_scan_source_file:
4001 0000A0A2 46                  <1>       inc    esi
4002 0000A0A3 803E20              <1>       cmp    byte [esi], 20h
4003 0000A0A6 7409                <1>       je     short rename_scan_destination_file_1
4004                              <1>       ;jb    short loc_rename_nofilename_retn
4005 0000A0A8 0F82D3EEFFFF        <1>       jb     loc_cmd_failed
4006 0000A0AE EBF2                <1>       jmp    short rename_scan_source_file
4007                              <1>
4008                              <1> loc_rename_nofilename_retn: ; 08/03/2016
4009 0000A0B0 C3                  <1>       retn
4010                              <1>
4011                              <1> rename_scan_destination_file_1:
4012 0000A0B1 C60600              <1>       mov    byte [esi], 0
4013                              <1>
4014                              <1> rename_scan_destination_file_2:
4015 0000A0B4 46                  <1>       inc    esi
4016 0000A0B5 803E20              <1>       cmp    byte [esi], 20h
4017 0000A0B8 74FA                <1>       je     short rename_scan_destination_file_2
4018                              <1>       ;jb    short loc_rename_nofilename_retn
4019 0000A0BA 0F82C1EEFFFF        <1>       jb     loc_cmd_failed
4020                              <1>
4021 0000A0C0 8935[C0940100]      <1>       mov    [DestinationFilePath], esi
4022                              <1>
4023                              <1> rename_scan_destination_file_3:
4024 0000A0C6 46                  <1>       inc    esi
4025 0000A0C7 803E20              <1>       cmp    byte [esi], 20h
4026 0000A0CA 77FA                <1>       ja     short rename_scan_destination_file_3
4027                              <1>
4028 0000A0CC C60600              <1>       mov    byte [esi], 0
4029                              <1>
4030                              <1> loc_rename_save_current_drive:
4031 0000A0CF 8A35[DE8A0100]      <1>       mov    dh, [Current_Drv]
4032 0000A0D5 8835[3A920100]      <1>       mov    byte [RUN_CDRV], dh
4033                              <1>
```

```
                                <1> loc_rename_sf_parse_path_name:
0000A0DB 8B35[BC940100]         <1>       mov   esi, [SourceFilePath]
0000A0E1 BF[7E930100]           <1>       mov   edi, FindFile_Drv
0000A0E6 E877150000             <1>       call  parse_path_name
0000A0EB 0F8290EEFFFF           <1>       jc    loc_cmd_failed
                                <1>
                                <1> loc_rename_sf_check_filename_exists:
0000A0F1 BE[C0930100]           <1>       mov   esi, FindFile_Name
0000A0F6 803E20                 <1>       cmp   byte [esi], 20h
0000A0F9 0F8682EEFFFF           <1>       jna   loc_cmd_failed
                                <1>
                                <1>       ;mov  [DelFile_FNPointer], esi
                                <1>
                                <1> loc_rename_sf_drv:
                                <1>       ;mov  dh, [Current_Drv]
                                <1>       ;mov  [RUN_CDRV], dh
                                <1>
0000A0FF 8A15[7E930100]         <1>       mov   dl, [FindFile_Drv]
0000A105 38F2                   <1>       cmp   dl, dh ; dh = [Current_Drv]
0000A107 740B                   <1>       je    short rename_sf_change_directory
                                <1>
0000A109 E8E0DFFFFF             <1>       call  change_current_drive
0000A10E 0F82ABF7FFFF           <1>       jc    loc_file_rw_cmd_failed
                                <1>
                                <1> rename_sf_change_directory:
0000A114 803D[7F930100]20       <1>       cmp   byte [FindFile_Directory], 20h
0000A11B 7618                   <1>       jna   short rename_sf_find
                                <1>
0000A11D FE05[A6400100]         <1>       inc   byte [Restore_CDIR]
0000A123 BE[7F930100]           <1>       mov   esi, FindFile_Directory
0000A128 30E4                   <1>       xor   ah, ah ; CD_COMMAND sign -> 0
0000A12A E81D0F0000             <1>       call  change_current_directory
0000A12F 0F828AF7FFFF           <1>       jc    loc_file_rw_cmd_failed
                                <1>
                                <1> ;rename_sf_change_prompt_dir_string:
                                <1>       ;call  change_prompt_dir_string
                                <1>
                                <1> rename_sf_find:
                                <1>       ;mov  esi, [DelFile_FNPointer]
0000A135 BE[C0930100]           <1>       mov   esi, FindFile_Name
                                <1>
0000A13A 66B80008               <1>       mov   ax, 0800h ; Except volume labels
0000A13E E84EF3FFFF             <1>       call  find_first_file
0000A143 0F8276F7FFFF           <1>       jc    loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rename_sf_ambgfn_check:
0000A149 6621D2                 <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
                                <1>       ;     (Note: It was BX in TRDOS v1)
                                <1>       ;jz   short loc_rename_sf_found
0000A14C 0F8571FCFFFF           <1>       jnz   loc_file_not_found
                                <1>
                                <1>       ;mov  eax, 2 ; File not found sign
                                <1>       ;stc
                                <1>       ;jmp  loc_file_rw_cmd_failed
                                <1>
                                <1> loc_rename_sf_found:
                                <1>       ; EDI = Directory buffer entry offset/address
                                <1>       ; BL = File (or Directory) Attributes
                                <1>       ;     (Note: It was 'CL' in TRDOS v1)
                                <1>       ; mov  bl, [EDI+0Bh]
                                <1>
0000A152 F6C307                 <1>       test  bl, 07h ; Attributes, S-H-R
0000A155 0F856EF7FFFF           <1>       jnz   loc_permission_denied
                                <1>
0000A15B BE[7E930100]           <1>         mov     esi, FindFile_Drv
0000A160 BF[C4940100]           <1>         mov     edi, SourceFile_Drv
0000A165 B920000000             <1>       mov   ecx, 32
0000A16A F3A5                   <1>       rep   movsd
                                <1>
                                <1> loc_rename_df_parse_path_name:
0000A16C 8B35[C0940100]         <1>       mov   esi, [DestinationFilePath]
0000A172 BF[7E930100]           <1>       mov   edi, FindFile_Drv
0000A177 E8E6140000             <1>       call  parse_path_name
0000A17C 7219                   <1>       jc    short loc_rename_df_cmd_failed
                                <1>
                                <1>       ;mov  dh, [RUN_CDRV]
0000A17E 8A35[DE8A0100]         <1>       mov   dh, [Current_Drv]
                                <1>
                                <1>       ; 'rename' command is valid only for same dos drive and same dir!
                                <1>       ; ('move' command must be used if source file and destination file
                                <1>       ; directories are not same!)
0000A184 8A15[7E930100]         <1>       mov   dl, [FindFile_Drv]
0000A18A 38F2                   <1>       cmp   dl, dh ; are source and destination drives different ?!
0000A18C 7509                   <1>       jne   short loc_rename_df_cmd_failed ; yes!
                                <1>
                                <1> rename_df_check_dirname_exists:
0000A18E 803D[7F930100]00       <1>       cmp   byte [FindFile_Directory], 0
0000A195 760B                   <1>       jna   short rename_df_check_filename_exists
                                <1>
                                <1>       ; different source file and destination file directories !
                                <1> loc_rename_df_cmd_failed:
0000A197 B801000000             <1>       mov   eax, 1 ; TRDOS 'Bad command or file name' error
0000A19C F9                     <1>       stc
0000A19D E91DF7FFFF             <1>       jmp   loc_file_rw_cmd_failed
                                <1>
                                <1> rename_df_check_filename_exists:
0000A1A2 BE[C0930100]           <1>       mov   esi, FindFile_Name
0000A1A7 E8A3F6FFFF             <1>       call  check_filename
0000A1AC 0F82BFF7FFFF           <1>       jc    loc_mkdir_invalid_dir_name_chars
                                <1>
                                <1>       ;mov  [DelFile_FNPointer], esi
                                <1>       ;cmp  byte [esi], 20h
                                <1>       ;ja   short loc_rename_df_find
                                <1>
                                <1>       ;mov  dh, [Current_Drv] ; dh has not been changed
```

```
                                <1>
                                <1> rename_df_drv_check_writable:
0000A1B2 0FB6F6                 <1>       movzx  esi, dh
                                <1>       ;movzx esi, byte [Current_Drv]
0000A1B5 81C600010900           <1>       add    esi, Logical_DOSDisks
                                <1>
0000A1BB 88F2                   <1>       mov    dl, dh ; dl = [Current_Drv]
0000A1BD 8A7601                 <1>       mov    dh, [esi+LD_DiskType]
                                <1>
0000A1C0 80FE01                 <1>       cmp    dh, 1 ; 0 = Invalid
0000A1C3 7310                   <1>       jnb    short rename_df_compare_sf_df_name
                                <1>
                                <1>       ; 16/10/2016 (13h -> 30)
0000A1C5 B81E000000             <1>       mov    eax, 30 ; 'Disk write-protected' error
0000A1CA 8B1D[C0940100]         <1>       mov    ebx, [DestinationFilePath]
0000A1D0 E9EAF6FFFF             <1>       jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> rename_df_compare_sf_df_name:
0000A1D5 BE[C0930100]           <1>       mov    esi, FindFile_Name
0000A1DA BF[06950100]           <1>       mov    edi, SourceFile_Name
0000A1DF B90C000000             <1>       mov    ecx, 12
                                <1> rename_df_compare_sf_df_name_next:
0000A1E4 AC                     <1>       lodsb
0000A1E5 AE                     <1>       scasb
0000A1E6 7506                   <1>       jne    short loc_rename_df_find
0000A1E8 08C0                   <1>       or     al, al
0000A1EA 74AB                   <1>       jz     short loc_rename_df_cmd_failed
0000A1EC E2F6                   <1>       loop   rename_df_compare_sf_df_name_next
                                <1>
                                <1> loc_rename_df_find:
                                <1>       ;mov   esi, [DelFile_FNPointer]
0000A1EE BE[C0930100]           <1>       mov    esi, FindFile_Name
                                <1>
0000A1F3 6631C0                 <1>       xor    ax, ax ; Any
0000A1F6 E896F2FFFF             <1>       call   find_first_file
                                <1>       ;jnc   short loc_rename_df_found
                                <1>       ; 29/12/2017
0000A1FB 0F83C8F6FFFF           <1>       jnc    loc_permission_denied
                                <1>
                                <1> loc_rename_df_check_error_code:
                                <1>       ;cmp   eax, 2
0000A201 3C02                   <1>       cmp    al, 2 ; Not found error
0000A203 7406                   <1>       je     short rename_df_move_find_struct_to_dest
0000A205 F9                     <1>       stc
0000A206 E9B4F6FFFF             <1>       jmp    loc_file_rw_cmd_failed
                                <1>
                                <1> ;loc_rename_df_found:
                                <1>       ; 05/11/2016
                                <1>       ; Permission denied error
                                <1>       ;mov   eax, ERR_PERM_DENIED ; 29/12/2017
                                <1>       ;stc
                                <1>       ;jmp   loc_permission_denied  ; 06/11/2016
                                <1>
                                <1> rename_df_move_find_struct_to_dest:
0000A20B BE[7E930100]           <1>         mov     esi, FindFile_Drv
0000A210 BF[44950100]           <1>         mov     edi, DestinationFile_Drv
0000A215 B920000000             <1>       mov    ecx, 32
0000A21A F3A5                   <1>       rep    movsd
                                <1>
                                <1> loc_rename_df_process_q_sf:
                                <1>       ;mov   ecx, 12
0000A21C B10C                   <1>       mov    cl, 12
0000A21E BE[06950100]           <1>       mov    esi, SourceFile_Name
0000A223 BF[F5450100]           <1>       mov    edi, Rename_OldName
                                <1> rename_df_process_q_nml_1_sf:
0000A228 AC                     <1>       lodsb
0000A229 3C20                   <1>         cmp  al, 20h
0000A22B 7603                   <1>         jna  short rename_df_process_q_nml_2_sf
0000A22D AA                     <1>       stosb
0000A22E E2F8                   <1>       loop   rename_df_process_q_nml_1_sf
                                <1>
                                <1> rename_df_process_q_nml_2_sf:
0000A230 C60700                 <1>       mov    byte [edi], 0
                                <1>
                                <1> loc_rename_df_process_q_df:
                                <1>       ;mov   ecx, 12
0000A233 B10C                   <1>       mov    cl, 12
0000A235 BE[86950100]           <1>       mov    esi, DestinationFile_Name
0000A23A BF[06460100]           <1>       mov    edi, Rename_NewName
                                <1> rename_df_process_q_nml_1_df:
0000A23F AC                     <1>       lodsb
0000A240 3C20                   <1>       cmp    al, 20h
0000A242 7603                   <1>       jna    short loc_rename_df_process_q_nml_2_df
0000A244 AA                     <1>       stosb
0000A245 E2F8                   <1>       loop   rename_df_process_q_nml_1_df
                                <1>
                                <1> loc_rename_df_process_q_nml_2_df:
0000A247 C60700                 <1>       mov    byte [edi], 0
                                <1>
                                <1> loc_rename_confirmation_question:
0000A24A BE[CD450100]           <1>       mov    esi, Msg_DoYouWantRename
0000A24F E811D3FFFF             <1>       call   print_msg
                                <1>
0000A254 A0[21950100]           <1>       mov    al, [SourceFile_DirEntry+11] ; Attributes
0000A259 2410                   <1>       and    al, 10h
0000A25B 750C                   <1>       jnz    short rename_confirmation_question_dir
                                <1>
                                <1> rename_confirmation_question_file:
0000A25D BE[E4450100]           <1>       mov    esi, Rename_File
0000A262 E8FED2FFFF             <1>       call   print_msg
0000A267 EB0A                   <1>       jmp    short rename_confirmation_question_as
                                <1>
                                <1> rename_confirmation_question_dir:
0000A269 BE[EA450100]           <1>       mov    esi, Rename_Directory
0000A26E E8F2D2FFFF             <1>       call   print_msg
```

```
                                <1>
                                <1> rename_confirmation_question_as:
0000A273 BE[F5450100]           <1>         mov    esi, Rename_OldName
0000A278 E8E8D2FFFF             <1>         call   print_msg
0000A27D BE[02460100]           <1>         mov    esi, Msg_File_rename_as
0000A282 E8DED2FFFF             <1>         call   print_msg
0000A287 BE[29450100]           <1>         mov    esi, Msg_YesNo
0000A28C E8D4D2FFFF             <1>         call   print_msg
                                <1>
                                <1> loc_rename_ask_again:
0000A291 30E4                   <1>         xor    ah, ah
0000A293 E8556CFFFF             <1>         call   int16h
0000A298 3C1B                   <1>         cmp    al, 1Bh
0000A29A 740F                   <1>         je     short loc_do_not_rename_file
0000A29C 24DF                   <1>         and    al, 0DFh
0000A29E A2[33450100]           <1>         mov    [Y_N_nextline], al
0000A2A3 3C59                   <1>         cmp    al, 'Y'
0000A2A5 7404                   <1>         je     short loc_yes_rename_file
0000A2A7 3C4E                   <1>         cmp    al, 'N'
0000A2A9 75E6                   <1>         jne    short loc_rename_ask_again
                                <1>
                                <1> loc_do_not_rename_file:
                                <1> loc_yes_rename_file:
0000A2AB E848F7FFFF             <1>         call   y_n_answer ; 29/12/2017
                                <1>         ;cmp   al, 'Y' ; 'yes'
                                <1>         ;cmc
                                <1>           ;jnc loc_file_rw_restore_retn
0000A2B0 3C4E                   <1>         cmp    al, 'N' ; 'no'
0000A2B2 0F8407F6FFFF           <1>           je   loc_file_rw_restore_retn
                                <1>
0000A2B8 BE[06460100]           <1>         mov    esi, Rename_NewName
0000A2BD 668B0D[3E950100]       <1>         mov    cx, [SourceFile_DirEntryNumber]
0000A2C4 66A1[2A950100]         <1>         mov    ax, [SourceFile_DirEntry+20] ; First Cluster, HW
0000A2CA C1E010                 <1>         shl    eax, 16 ; 13/11/2017
0000A2CD 66A1[30950100]         <1>         mov    ax, [SourceFile_DirEntry+26] ; First Cluster, LW
                                <1>
0000A2D3 0FB61D[13950100]       <1>         movzx  ebx, byte [SourceFile_LongNameEntryLength]
0000A2DA E8CB1B0000             <1>         call   rename_directory_entry
0000A2DF E9FBF6FFFF             <1>         jmp    loc_rename_file_ok
                                <1> ;loc_rename_file_ok:
                                <1> ;       jc     loc_run_cmd_failed
                                <1> ;       mov    esi, Msg_OK
                                <1> ;       call   proc_printmsg
                                <1> ;       jmp    loc_file_rw_restore_retn
                                <1>
                                <1> move_file:
                                <1>         ; 11/03/2016
                                <1>         ; 09/03/2016
                                <1>         ; 08/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_move')
                                <1>         ; 23/04/2011
                                <1>
                                <1> get_move_source_fchar:
                                <1>         ; esi = file name
0000A2E4 803E20                 <1>         cmp    byte [esi], 20h
0000A2E7 7614                   <1>           jna  short loc_move_nofilename_retn
                                <1>
0000A2E9 8935[BC940100]         <1>         mov    [SourceFilePath], esi
                                <1>
                                <1> move_scan_source_file:
0000A2EF 46                     <1>         inc    esi
0000A2F0 803E20                 <1>         cmp    byte [esi], 20h
0000A2F3 7409                   <1>           je      short move_scan_destination_1
                                <1>         ;jb    short loc_move_nofilename_retn
0000A2F5 0F8286ECFFFF           <1>         jb     loc_cmd_failed
0000A2FB EBF2                   <1>         jmp    short move_scan_source_file
                                <1>
                                <1> loc_move_nofilename_retn:
0000A2FD C3                     <1>         retn
                                <1>
                                <1> move_scan_destination_1:
0000A2FE C60600                 <1>         mov    byte [esi], 0
                                <1>
                                <1> move_scan_destination_2:
0000A301 46                     <1>         inc    esi
0000A302 803E20                 <1>         cmp    byte [esi], 20h
0000A305 74FA                   <1>         je     short move_scan_destination_2
                                <1>         ;jb    short loc_move_nofilename_retn
0000A307 0F8274ECFFFF           <1>         jb     loc_cmd_failed
                                <1>
0000A30D 8935[C0940100]         <1>         mov    [DestinationFilePath], esi
                                <1>
                                <1> move_scan_destination_3:
0000A313 46                     <1>         inc    esi
0000A314 803E20                 <1>         cmp    byte [esi], 20h
0000A317 77FA                   <1>         ja     short move_scan_destination_3
0000A319 C60600                 <1>         mov    byte [esi], 0
                                <1>
                                <1> loc_move_scan_destination_OK:
0000A31C 8B35[BC940100]         <1>         mov    esi, [SourceFilePath]
0000A322 8B3D[C0940100]         <1>         mov    edi, [DestinationFilePath]
                                <1>
0000A328 B001                   <1>         mov    al, 1  ; move procedure Phase 1
0000A32A E8F71B0000             <1>         call   move_source_file_to_destination_file
0000A32F 7328                   <1>         jnc    short move_source_file_to_destination_question
                                <1>
                                <1> loc_move_cmd_failed_1:
0000A331 08C0                   <1>         or     al, al
0000A333 0F8448ECFFFF           <1>         jz     loc_cmd_failed
0000A339 3C11                   <1>         cmp    al, 11h
0000A33B 740D                   <1>         je     short loc_msg_not_same_device
                                <1>         ;cmp   al, 05h
                                <1>         ;cmp   al, ERR_PERM_DENIED ; 29/12/2017
                                <1>         ;jne   loc_run_cmd_failed
                                <1>         ;jmp   loc_permission_denied
```

```
4349 0000A33D 3C0B              <1>       cmp   al, ERR_PERM_DENIED
4350 0000A33F 0F8484F5FFFF      <1>       je    loc_permission_denied
4351 0000A345 E962ECFFFF        <1>       jmp   loc_run_cmd_failed
4352                            <1>
4353                            <1>       ;mov  esi, Msg_Permission_denied
4354                            <1>       ;call print_msg
4355                            <1>       ;jmp  loc_file_rw_restore_retn
4356                            <1>
4357                            <1> loc_msg_not_same_device:
4358 0000A34A BE[13460100]      <1>       mov   esi, msg_not_same_drv
4359 0000A34F E811D2FFFF        <1>       call  print_msg
4360 0000A354 E966F5FFFF        <1>       jmp   loc_file_rw_restore_retn
4361                            <1>
4362                            <1> move_source_file_to_destination_question:
4363 0000A359 A0[C4940100]      <1>         mov     al, [SourceFile_Drv]
4364 0000A35E 0441              <1>       add   al, 'A'
4365 0000A360 A2[75460100]      <1>       mov   [msg_source_file_drv], al
4366 0000A365 A0[44950100]      <1>         mov     al, [DestinationFile_Drv]
4367 0000A36A 0441              <1>       add   al, 'A'
4368 0000A36C A2[94460100]      <1>       mov   [msg_destination_file_drv], al
4369                            <1>
4370 0000A371 57                <1>       push  edi ; *
4371                            <1>
4372 0000A372 BE[59460100]      <1>       mov   esi, msg_source_file
4373 0000A377 E8E9D1FFFF        <1>       call  print_msg
4374 0000A37C BE[C5940100]      <1>       mov   esi, SourceFile_Directory
4375 0000A381 803E20            <1>       cmp   byte [esi], 20h
4376 0000A384 7605              <1>       jna   short msftdfq_sfn
4377 0000A386 E8DAD1FFFF        <1>       call  print_msg
4378                            <1> msftdfq_sfn:
4379 0000A38B BE[06950100]      <1>       mov   esi, SourceFile_Name
4380 0000A390 E8D0D1FFFF        <1>       call  print_msg
4381 0000A395 BE[78460100]      <1>       mov   esi, msg_destination_file
4382 0000A39A E8C6D1FFFF        <1>       call  print_msg
4383 0000A39F BE[45950100]      <1>       mov   esi, DestinationFile_Directory
4384 0000A3A4 803E20            <1>       cmp   byte [esi], 20h
4385 0000A3A7 7605              <1>       jna   short msftdfq_dfn
4386 0000A3A9 E8B7D1FFFF        <1>       call  print_msg
4387                            <1> msftdfq_dfn:
4388 0000A3AE BE[86950100]      <1>       mov   esi, DestinationFile_Name
4389 0000A3B3 E8ADD1FFFF        <1>       call  print_msg
4390 0000A3B8 BE[97460100]      <1>       mov   esi, msg_copy_nextline
4391 0000A3BD E8A3D1FFFF        <1>       call  print_msg
4392 0000A3C2 BE[97460100]      <1>       mov   esi, msg_copy_nextline
4393 0000A3C7 E899D1FFFF        <1>       call  print_msg
4394                            <1>
4395                            <1> loc_move_ask_for_new_file_yes_no:
4396 0000A3CC BE[25460100]      <1>       mov   esi, Msg_DoYouWantMoveFile
4397 0000A3D1 E88FD1FFFF        <1>       call  print_msg
4398 0000A3D6 BE[29450100]      <1>       mov   esi, Msg_YesNo
4399 0000A3DB E885D1FFFF        <1>       call  print_msg
4400                            <1> loc_move_ask_for_new_file_again:
4401 0000A3E0 30E4              <1>       xor   ah, ah
4402 0000A3E2 E8066BFFFF        <1>       call  int16h
4403 0000A3E7 3C1B              <1>       cmp   al, 1Bh
4404                            <1>       ;je   short loc_do_not_move_file
4405 0000A3E9 7441              <1>       je    short loc_move_y_n_escape
4406 0000A3EB 24DF              <1>       and   al, 0DFh
4407 0000A3ED A2[33450100]      <1>         mov     [Y_N_nextline], al
4408 0000A3F2 3C59              <1>       cmp   al, 'Y'
4409 0000A3F4 7404              <1>       je    short loc_yes_move_file
4410 0000A3F6 3C4E              <1>       cmp   al, 'N'
4411 0000A3F8 75E6              <1>       jne   short loc_move_ask_for_new_file_again
4412                            <1>
4413                            <1> loc_do_not_move_file:
4414                            <1> loc_yes_move_file:
4415 0000A3FA E8F9F5FFFF        <1>       call  y_n_answer ; 29/12/2017
4416 0000A3FF 5F                <1>       pop   edi ; *
4417                            <1>       ;cmp  al, 'Y' ; 'yes'
4418                            <1>       ;cmc
4419                            <1>         ;jnc loc_file_rw_restore_retn
4420 0000A400 3C4E              <1>       cmp   al, 'N' ; 'no'
4421 0000A402 0F84B7F4FFFF      <1>         je  loc_file_rw_restore_retn
4422                            <1>
4423                            <1> loc_move_yes_move_file:
4424 0000A408 B002              <1>       mov   al, 2 ; move procedure Phase 2
4425 0000A40A E8171B0000        <1>       call  move_source_file_to_destination_file
4426                            <1>       ;jc   short loc_move_cmd_failed_2
4427 0000A40F 0F83D0F5FFFF      <1>         jnc     move_source_file_to_destination_OK
4428                            <1>
4429                            <1> ;move_source_file_to_destination_OK:
4430                            <1> ;     mov   esi, Msg_OK
4431                            <1> ;     call  print_msg
4432                            <1> ;     jmp   loc_file_rw_restore_retn
4433                            <1>
4434                            <1> loc_move_cmd_failed_2:
4435 0000A415 3C27              <1>       cmp   al, 27h
4436 0000A417 0F858FEBFFFF      <1>       jne   loc_run_cmd_failed
4437                            <1>
4438 0000A41D BE[3E460100]      <1>       mov   esi, msg_insufficient_disk_space
4439 0000A422 E83ED1FFFF        <1>       call  print_msg
4440                            <1>
4441 0000A427 E993F4FFFF        <1>       jmp   loc_file_rw_restore_retn
4442                            <1>
4443                            <1> loc_move_y_n_escape:
4444 0000A42C B04E              <1>       mov   al, 'N' ; 'no'
4445 0000A42E EBCA              <1>       jmp   short loc_do_not_move_file
4446                            <1>
4447                            <1> copy_file:
4448                            <1>       ; 15/10/2016
4449                            <1>       ; 24/03/2016
4450                            <1>       ; 21/03/2016
4451                            <1>       ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
4452                            <1>       ; 21/05/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_copy')
4453                            <1>       ; 01/08/2010
```

```
                                <1>
                                <1> get_copy_source_fchar:
                                <1>       ; esi = file name
0000A430 803E20                 <1>       cmp    byte [esi], 20h
0000A433 7614                   <1>         jna     short loc_copy_nofilename_retn
                                <1>
0000A435 8935[BC940100]         <1>       mov    [SourceFilePath], esi
                                <1>
                                <1> copy_scan_source_file:
0000A43B 46                     <1>       inc    esi
0000A43C 803E20                 <1>       cmp    byte [esi], 20h
0000A43F 7409                   <1>       je     short copy_scan_destination_1
                                <1>       ;jb    short loc_copy_nofilename_retn
0000A441 0F823AEBFFFF           <1>       jb     loc_cmd_failed
0000A447 EBF2                   <1>       jmp    short copy_scan_source_file
                                <1>
                                <1> loc_copy_nofilename_retn:
0000A449 C3                     <1>       retn
                                <1>
                                <1> copy_scan_destination_1:
0000A44A C60600                 <1>       mov    byte [esi], 0
                                <1>
                                <1> copy_scan_destination_2:
0000A44D 46                     <1>       inc    esi
0000A44E 803E20                 <1>       cmp    byte [esi], 20h
0000A451 74FA                   <1>       je     short copy_scan_destination_2
                                <1>       ;jb    short loc_copy_nofilename_retn
0000A453 0F8228EBFFFF           <1>       jb     loc_cmd_failed
                                <1>
0000A459 8935[C0940100]         <1>       mov    [DestinationFilePath], esi
                                <1>
                                <1> copy_scan_destination_3:
0000A45F 46                     <1>       inc    esi
0000A460 803E20                 <1>       cmp    byte [esi], 20h
0000A463 77FA                   <1>       ja     short copy_scan_destination_3
0000A465 C60600                 <1>       mov    byte [esi], 0
                                <1>
                                <1> loc_copy_save_current_drive:
0000A468 8A35[DE8A0100]         <1>       mov    dh, [Current_Drv]
0000A46E 8835[3A920100]         <1>       mov    [RUN_CDRV], dh
                                <1>
                                <1> copy_source_file_to_destination_phase_1:
0000A474 8B35[BC940100]         <1>       mov    esi, [SourceFilePath]
0000A47A 8B3D[C0940100]         <1>       mov    edi, [DestinationFilePath]
                                <1>
0000A480 B001                   <1>       mov    al, 1  ; copy procedure Phase 1
0000A482 E83C1D0000             <1>       call   copy_source_file_to_destination_file
0000A487 732B                   <1>       jnc    short copy_source_file_to_destination_question
                                <1>
                                <1> loc_copy_cmd_failed_1:
                                <1>       ; 18/03/2016 (restore current drive and directory)
0000A489 08C0                   <1>       or     al, al
0000A48B 7507                   <1>       jnz    short loc_copy_cmd_failed_2
                                <1>
0000A48D FEC0                   <1>         inc     al ; mov al, 1 ; Bad command or file name !
0000A48F E918EBFFFF             <1>       jmp    loc_run_cmd_failed
                                <1>
                                <1> loc_copy_cmd_failed_2:
0000A494 3C27                   <1>       cmp    al, 27h ; Insufficient disk space
0000A496 740D                   <1>       je     short loc_file_write_insuff_disk_space_msg
                                <1>
                                <1>       ; 29/12/2017
                                <1>       ;cmp   al, 05h
0000A498 3C0B                   <1>       cmp    al, ERR_PERM_DENIED
0000A49A 0F850CEBFFFF           <1>       jne    loc_run_cmd_failed
                                <1>
0000A4A0 E924F4FFFF             <1>       jmp    loc_permission_denied
                                <1>
                                <1> loc_file_write_insuff_disk_space_msg:
0000A4A5 BE[3E460100]           <1>       mov    esi, msg_insufficient_disk_space
0000A4AA E8B6D0FFFF             <1>       call   print_msg
0000A4AF E90BF4FFFF             <1>         jmp     loc_file_rw_restore_retn
                                <1>
                                <1> copy_source_file_to_destination_question:
0000A4B4 57                     <1>       push   edi ; *
                                <1>
                                <1>       ; dh = source file attributes
                                <1>       ; dl > 0 -> destination file found
0000A4B5 20D2                   <1>       and    dl, dl
0000A4B7 7449                   <1>       jz     short copy_source_file_to_destination_pass_owrq
                                <1>
                                <1> loc_copy_ask_for_owr_yes_no:
0000A4B9 BE[9A460100]           <1>       mov    esi, Msg_DoYouWantOverWriteFile
0000A4BE E8A2D0FFFF             <1>       call   print_msg
0000A4C3 BE[86950100]           <1>       mov    esi, DestinationFile_Name
0000A4C8 E898D0FFFF             <1>       call   print_msg
0000A4CD BE[29450100]           <1>       mov    esi, Msg_YesNo
0000A4D2 E88ED0FFFF             <1>       call   print_msg
                                <1>
                                <1> loc_copy_ask_for_owr_again:
0000A4D7 30E4                   <1>       xor    ah, ah
0000A4D9 E80F6AFFFF             <1>       call   int16h
0000A4DE 3C1B                   <1>       cmp    al, 1Bh
                                <1>         ;je     loc_do_not_copy_file
0000A4E0 7419                   <1>         je      short loc_copy_y_n_escape
0000A4E2 24DF                   <1>       and    al, 0DFh
0000A4E4 A2[33450100]           <1>         mov     [Y_N_nextline], al
0000A4E9 3C59                   <1>       cmp    al, 'Y'
0000A4EB 0F84B1000000           <1>         je      loc_yes_copy_file
0000A4F1 3C4E                   <1>       cmp    al, 'N'
0000A4F3 0F84A9000000           <1>         je      loc_do_not_copy_file
0000A4F9 EBDC                   <1>       jmp    short loc_copy_ask_for_owr_again
                                <1>
                                <1> loc_copy_y_n_escape:
0000A4FB B04E                   <1>       mov    al, 'N' ; 'no'
```

```
4559 0000A4FD E9A0000000          <1>         jmp     loc_do_not_copy_file
4560                              <1>
4561                              <1> copy_source_file_to_destination_pass_owrq:
4562 0000A502 A0[C4940100]        <1>         mov     al, [SourceFile_Drv]
4563 0000A507 0441                <1>         add     al, 'A'
4564 0000A509 A2[75460100]        <1>         mov     [msg_source_file_drv], al
4565 0000A50E A0[44950100]        <1>         mov     al, [DestinationFile_Drv]
4566 0000A513 0441                <1>         add     al, 'A'
4567 0000A515 A2[94460100]        <1>         mov     [msg_destination_file_drv], al
4568                              <1>
4569 0000A51A BE[59460100]        <1>         mov     esi, msg_source_file
4570 0000A51F E841D0FFFF          <1>         call    print_msg
4571 0000A524 BE[C5940100]        <1>         mov     esi, SourceFile_Directory
4572 0000A529 803E20              <1>         cmp     byte [esi], 20h
4573 0000A52C 7605                <1>         jna     short csftdfq_sfn
4574 0000A52E E832D0FFFF          <1>         call    print_msg
4575                              <1> csftdfq_sfn:
4576 0000A533 BE[06950100]        <1>         mov     esi, SourceFile_Name
4577 0000A538 E828D0FFFF          <1>         call    print_msg
4578 0000A53D BE[78460100]        <1>         mov     esi, msg_destination_file
4579 0000A542 E81ED0FFFF          <1>         call    print_msg
4580 0000A547 BE[45950100]        <1>         mov     esi, DestinationFile_Directory
4581 0000A54C 803E20              <1>         cmp     byte [esi], 20h
4582 0000A54F 7605                <1>         jna     short csftdfq_dfn
4583 0000A551 E80FD0FFFF          <1>         call    print_msg
4584                              <1> csftdfq_dfn:
4585 0000A556 BE[86950100]        <1>         mov     esi, DestinationFile_Name
4586 0000A55B E805D0FFFF          <1>         call    print_msg
4587 0000A560 BE[97460100]        <1>         mov     esi, msg_copy_nextline
4588 0000A565 E8FBCFFFFF          <1>         call    print_msg
4589 0000A56A BE[97460100]        <1>         mov     esi, msg_copy_nextline
4590 0000A56F E8F1CFFFFF          <1>         call    print_msg
4591                              <1>
4592                              <1> loc_copy_ask_for_new_file_yes_no:
4593 0000A574 BE[B9460100]        <1>         mov     esi, Msg_DoYouWantCopyFile
4594 0000A579 E8E7CFFFFF          <1>         call    print_msg
4595 0000A57E BE[29450100]        <1>         mov     esi, Msg_YesNo
4596 0000A583 E8DDCFFFFF          <1>         call    print_msg
4597                              <1>
4598                              <1> loc_copy_ask_for_new_file_again:
4599 0000A588 30E4                <1>         xor     ah, ah
4600 0000A58A E85E69FFFF          <1>         call    int16h
4601 0000A58F 3C1B                <1>         cmp     al, 1Bh
4602 0000A591 740F                <1>         je      short loc_do_not_copy_file
4603 0000A593 24DF                <1>         and     al, 0DFh
4604 0000A595 A2[33450100]        <1>         mov     [Y_N_nextline], al
4605 0000A59A 3C59                <1>         cmp     al, 'Y'
4606 0000A59C 7404                <1>         je      short loc_yes_copy_file
4607 0000A59E 3C4E                <1>         cmp     al, 'N'
4608 0000A5A0 75E6                <1>         jne     short loc_copy_ask_for_new_file_again
4609                              <1>
4610                              <1> loc_do_not_copy_file:
4611                              <1> loc_yes_copy_file:
4612 0000A5A2 E851F4FFFF          <1>         call    y_n_answer ; 29/12/2017
4613 0000A5A7 5F                  <1>         pop     edi ; *
4614                              <1>         ;cmp    al, 'Y' ; 'yes'
4615                              <1>         ;cmc
4616                              <1>         ;jnc    loc_file_rw_restore_retn
4617 0000A5A8 3C4E                <1>         cmp     al, 'N' ; 'no'
4618 0000A5AA 0F840FF3FFFF        <1>         je      loc_file_rw_restore_retn
4619                              <1>
4620                              <1> copy_source_file_to_destination_pass_q:
4621 0000A5B0 B002                <1>         mov     al, 2  ; copy procedure Phase 2
4622 0000A5B2 E80C1C0000          <1>         call    copy_source_file_to_destination_file
4623                              <1>         ;jc     short loc_file_write_check_disk_space_err
4624                              <1>
4625                              <1>         ; 24/03/2016
4626                              <1>         ;push   cx
4627 0000A5B7 51                  <1>         push    ecx ; 29/12/2017
4628 0000A5B8 BE[97460100]        <1>         mov     esi, msg_copy_nextline
4629 0000A5BD E8A3CFFFFF          <1>         call    print_msg
4630 0000A5C2 58                  <1>         pop     eax ; 29/12/2017
4631                              <1>         ;;pop   cx
4632                              <1>         ;pop    ax
4633                              <1>
4634                              <1>         ;or     cl, cl
4635 0000A5C3 08C0                <1>         or      al, al
4636 0000A5C5 7419                <1>         jz      short copy_source_file_to_destination_OK
4637                              <1>
4638                              <1>         ; 15/10/2016 (1Dh -> 18)
4639                              <1>         ; 18/03/2016 (1Dh)
4640                              <1>         ;cmp    cl, 18 ; write error
4641 0000A5C7 3C12                <1>         cmp     al, 18
4642 0000A5C9 7506                <1>         jne     short copy_source_file_to_destination_not_OK
4643                              <1>         ;
4644                              <1>         ;mov    al, cl ; error number (write fault!)
4645 0000A5CB F9                  <1>         stc
4646 0000A5CC E9EEF2FFFF          <1>         jmp     loc_file_rw_cmd_failed
4647                              <1>
4648                              <1> copy_source_file_to_destination_not_OK:
4649 0000A5D1 BE[D2460100]        <1>         mov     esi, Msg_read_file_error_before_EOF
4650 0000A5D6 E88ACFFFFF          <1>         call    print_msg
4651 0000A5DB E9DFF2FFFF          <1>         jmp     loc_file_rw_restore_retn
4652                              <1>
4653                              <1> copy_source_file_to_destination_OK:
4654 0000A5E0 BE[37450100]        <1>         mov     esi, Msg_OK
4655 0000A5E5 E87BCFFFFF          <1>         call    print_msg
4656                              <1>
4657 0000A5EA E9D0F2FFFF          <1>         jmp     loc_file_rw_restore_retn
4658                              <1>
4659                              <1> ;loc_file_write_check_disk_space_err:
4660                              <1>         ;cmp    al, 27h ; Insufficient disk space
4661                              <1>         ;je     loc_file_write_insuff_disk_space_msg
4662                              <1>         ;jb     loc_file_rw_cmd_failed
4663                              <1>
```

```
4664                                    <1>       ;call  print_misc_error_msg ; 15/03/2016
4665                                    <1>         ;jmp loc_file_rw_restore_retn
4666                                    <1>
4667                                    <1> change_fs_file_attributes:
4668                                    <1>       ; 04/03/2016 ; Temporary
4669                                    <1>       ; AL = File or directory attributes
4670                                    <1>       ; AH = 0 -> Attributes are in MS-DOS format
4671                                    <1>       ; AH > 0 -> Attributes are in SINGLIX format
4672                                    <1>       ;push  ebx
4673                                    <1>       ; ... do somethings here ...
4674                                    <1>       ;pop   ebx
4675                                    <1>       ; BL = File or directory attributes
4676 0000A5EF C3                        <1>       retn
4677                                    <1>
4678                                    <1> set_get_env:
4679                                    <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
4680                                    <1>       ; 02/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_set')
4681                                    <1>       ; 2005 - 28/08/2011
4682                                    <1> get_setenv_fchar:
4683                                    <1>       ; esi = environment variable/string
4684 0000A5F0 8A06                     <1>       mov    al, [esi]
4685 0000A5F2 3C20                     <1>       cmp    al, 20h
4686 0000A5F4 771E                     <1>       ja     short loc_find_env
4687                                    <1>
4688 0000A5F6 BE00300900               <1>       mov    esi, Env_Page
4689                                    <1> loc_print_setline:
4690 0000A5FB 803E00                   <1>       cmp    byte [esi], 0
4691 0000A5FE 7613                     <1>       jna    short loc_setenv_retn
4692 0000A600 E860CFFFFF               <1>       call   print_msg
4693 0000A605 56                       <1>       push   esi
4694 0000A606 BE[434D0100]             <1>       mov    esi, nextline
4695 0000A60B E855CFFFFF               <1>       call   print_msg
4696 0000A610 5E                       <1>       pop    esi
4697 0000A611 EBE8                     <1>       jmp    short loc_print_setline
4698                                    <1>
4699                                    <1> loc_setenv_retn:
4700 0000A613 C3                       <1>       retn
4701                                    <1>
4702                                    <1> loc_find_env:
4703 0000A614 3C3D                     <1>       cmp    al, '='
4704 0000A616 0F8465E9FFFF             <1>       je     loc_cmd_failed
4705                                    <1>
4706 0000A61C 56                       <1>       push   esi
4707                                    <1> loc_repeat_env_equal_check:
4708 0000A61D 46                       <1>       inc    esi
4709 0000A61E 803E3D                   <1>       cmp    byte [esi], '='
4710 0000A621 7431                     <1>       je     short pass_env_equal_check
4711 0000A623 803E20                   <1>       cmp    byte [esi], 20h
4712 0000A626 73F5                     <1>       jnb    short loc_repeat_env_equal_check
4713 0000A628 C60600                   <1>       mov    byte [esi], 0
4714 0000A62B 5E                       <1>       pop    esi
4715 0000A62C BF[DE8B0100]             <1>       mov    edi, TextBuffer ; out buffer
4716 0000A631 B9FF000000               <1>       mov    ecx, 255 ; maximum size (limit)
4717 0000A636 30C0                     <1>       xor    al, al ; 0 -> use [ESI]
4718 0000A638 E89E000000               <1>       call   get_environment_string
4719 0000A63D 72D4                     <1>       jc     short loc_setenv_retn
4720                                    <1>
4721 0000A63F BE[DE8B0100]             <1>       mov    esi, TextBuffer
4722 0000A644 E81CCFFFFF               <1>       call   print_msg
4723 0000A649 BE[434D0100]             <1>       mov    esi, nextline
4724 0000A64E E812CFFFFF               <1>       call   print_msg
4725                                    <1>
4726 0000A653 C3                       <1>       retn
4727                                    <1>
4728                                    <1> pass_env_equal_check:
4729 0000A654 46                       <1>       inc    esi
4730 0000A655 803E20                   <1>       cmp    byte [esi], 20h
4731 0000A658 73FA                     <1>       jnb    short pass_env_equal_check
4732 0000A65A C60600                   <1>       mov    byte [esi], 0
4733                                    <1>
4734                                    <1> loc_call_set_env_string:
4735 0000A65D 5E                       <1>       pop    esi
4736 0000A65E E83B010000               <1>       call   set_environment_string
4737 0000A663 73AE                     <1>       jnc    short loc_setenv_retn
4738                                    <1>
4739                                    <1> loc_set_cmd_failed:
4740 0000A665 3C08                     <1>       cmp    al, 08h
4741 0000A667 0F8514E9FFFF             <1>       jne    loc_cmd_failed
4742                                    <1>
4743 0000A66D BE[12470100]             <1>       mov    esi, Msg_No_Set_Space
4744 0000A672 E8EECEFFFF               <1>       call   print_msg
4745                                    <1>
4746 0000A677 C3                       <1>       retn
4747                                    <1>
4748                                    <1> set_get_path:
4749                                    <1>       ; 11/04/2016 (TRDOS 386 = TRDOS v2.0)
4750                                    <1>       ; 03/09/2011 (TRDOS v1, CMD_INTR.ASM, 'cmp_cmd_path')
4751                                    <1>       ; 2005
4752                                    <1> get_path_fchar:
4753                                    <1>       ; esi = path
4754 0000A678 803E20                   <1>       cmp    byte [esi], 20h
4755 0000A67B 7737                     <1>       ja     short loc_set_path
4756                                    <1>
4757 0000A67D BE00300900               <1>       mov    esi, Env_Page
4758                                    <1> loc_print_path:
4759 0000A682 803E00                   <1>       cmp    byte [esi], 0
4760 0000A685 762C                     <1>       jna    short loc_path_retn
4761                                    <1>
4762 0000A687 BE[72410100]             <1>       mov    esi, Cmd_Path ; 'PATH' address
4763 0000A68C BF[DE8B0100]             <1>       mov    edi, TextBuffer ; out buffer
4764 0000A691 30C0                     <1>       xor    al, al  ; use [ESI]
4765 0000A693 B9FF000000               <1>       mov    ecx, 255 ; maximum size (limit)
4766 0000A698 E83E000000               <1>       call   get_environment_string
4767 0000A69D 7214                     <1>       jc     short loc_path_retn
4768                                    <1>
```

```
4769 0000A69F BE[DE8B0100]        <1>       mov   esi, TextBuffer
4770 0000A6A4 E8BCCEFFFF          <1>       call  print_msg
4771 0000A6A9 BE[434D0100]        <1>       mov   esi, nextline
4772 0000A6AE E8B2CEFFFF          <1>       call  print_msg
4773                              <1>
4774                              <1> loc_path_retn:
4775 0000A6B3 C3                  <1>       retn
4776                              <1>
4777                              <1> loc_set_path:
4778 0000A6B4 56                  <1>       push  esi
4779                              <1> loc_set_path_find_end:
4780 0000A6B5 46                  <1>       inc   esi
4781 0000A6B6 803E20              <1>       cmp   byte [esi], 20h
4782 0000A6B9 73FA                <1>       jnb   short loc_set_path_find_end
4783 0000A6BB C60600              <1>       mov   byte [esi], 0
4784                              <1> loc_set_path_header:
4785 0000A6BE 5E                  <1>       pop   esi
4786                              <1> set_path_x: ; 31/12/2017 ('syspath')
4787 0000A6BF 4E                  <1>       dec   esi
4788 0000A6C0 C6063D              <1>       mov   byte [esi], '='
4789 0000A6C3 4E                  <1>       dec   esi
4790 0000A6C4 C60648              <1>       mov   byte [esi], 'H'
4791 0000A6C7 4E                  <1>       dec   esi
4792 0000A6C8 C60654              <1>       mov   byte [esi], 'T'
4793 0000A6CB 4E                  <1>       dec   esi
4794 0000A6CC C60641              <1>       mov   byte [esi], 'A'
4795 0000A6CF 4E                  <1>       dec   esi
4796 0000A6D0 C60650              <1>       mov   byte [esi], 'P'
4797                              <1>
4798                              <1> loc_path_call_set_env_string:
4799 0000A6D3 E8C6000000          <1>       call  set_environment_string
4800 0000A6D8 728B                <1>         jc  short loc_set_cmd_failed
4801                              <1>
4802 0000A6DA C3                  <1>       retn
4803                              <1>
4804                              <1> get_environment_string:
4805                              <1>       ; 12/04/2016
4806                              <1>       ; 11/04/2016
4807                              <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
4808                              <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
4809                              <1>       ; 28/08/2011
4810                              <1>       ; INPUT->
4811                              <1>       ;     EDI = Output buffer
4812                              <1>       ;     CX = Buffer length (<= ENV_PAGE_SIZE)
4813                              <1>       ;
4814                              <1>       ;     AL > 0 = AL = String sequence number
4815                              <1>       ;     AL = 0 -> ESI = ASCIIZ Set word
4816                              <1>       ;           (environment variable)
4817                              <1>       ; OUTPUT ->
4818                              <1>       ;     ESI is not changed
4819                              <1>       ;     EDI is not changed
4820                              <1>       ;     EAX = String length (with zero tail)
4821                              <1>       ;     EDX = Environment variables page address
4822                              <1>       ;     CF = 1 -> Not found (EAX not valid)
4823                              <1>       ;
4824                              <1>       ; (Modified registers: EAX, EDX)
4825                              <1>
4826 0000A6DB BA00300900          <1>       mov   edx, Env_Page
4827 0000A6E0 803A00              <1>       cmp   byte [edx], 0
4828 0000A6E3 7474                <1>       jz    short get_env_string_with_word_stc_retn
4829                              <1>
4830 0000A6E5 66890D[48960100]    <1>       mov   [env_var_length], cx
4831                              <1>
4832 0000A6EC 51                  <1>       push  ecx ; *
4833 0000A6ED 56                  <1>       push  esi ; **
4834                              <1>
4835 0000A6EE 08C0                <1>       or    al, al
4836 0000A6F0 7449                <1>       jz    short get_env_string_with_word
4837                              <1>
4838                              <1> get_env_string_with_seq_number:
4839 0000A6F2 B101                <1>       mov   cl, 1
4840 0000A6F4 88C5                <1>       mov   ch, al
4841 0000A6F6 31C0                <1>       xor   eax, eax
4842 0000A6F8 89D6                <1>       mov   esi, edx ; Env_Page
4843                              <1>
4844                              <1> get_env_string_seq_number_check:
4845 0000A6FA 38CD                <1>       cmp   ch, cl
4846 0000A6FC 7726                <1>       ja    short get_env_string_seq_number_next
4847                              <1>
4848                              <1> get_env_string_move_to_buff:
4849 0000A6FE 57                  <1>       push  edi ; ***
4850                              <1>
4851 0000A6FF 29D2                <1>       sub   edx, edx
4852                              <1>
4853                              <1> get_env_string_seq_number_repeat1:
4854 0000A701 42                  <1>       inc   edx
4855 0000A702 AC                  <1>       lodsb
4856 0000A703 AA                  <1>       stosb
4857                              <1>
4858 0000A704 66FF0D[48960100]    <1>       dec   word [env_var_length]
4859 0000A70B 7508                <1>       jnz   short get_env_string_seq_number_repeat3
4860                              <1>
4861                              <1> get_env_string_seq_number_repeat2:
4862 0000A70D 20C0                <1>       and   al, al
4863 0000A70F 7408                <1>       jz    short get_env_string_seq_number_ok
4864 0000A711 42                  <1>       inc   edx
4865 0000A712 AC                  <1>       lodsb
4866 0000A713 EBF8                <1>       jmp   short get_env_string_seq_number_repeat2
4867                              <1>
4868                              <1> get_env_string_seq_number_repeat3:
4869 0000A715 08C0                <1>       or    al, al
4870 0000A717 75E8                <1>       jnz   short get_env_string_seq_number_repeat1
4871                              <1>
4872                              <1> get_env_string_seq_number_ok:
4873 0000A719 5F                  <1>       pop   edi ; ***
```

```
4874 0000A71A 89D0              <1>       mov    eax, edx ; Length of the environment string
4875                            <1>                       ; (ASCIIZ, includes ZERO tail)
4876 0000A71C BA00300900        <1>       mov    edx, Env_Page
4877                            <1>
4878                            <1> get_env_string_stc_retn:
4879 0000A721 5E                <1>       pop    esi ; **
4880 0000A722 59                <1>       pop    ecx ; *
4881 0000A723 C3                <1>       retn
4882                            <1>
4883                            <1> get_env_string_seq_number_next:
4884 0000A724 AC                <1>       lodsb
4885 0000A725 08C0              <1>       or     al, al
4886 0000A727 75FB              <1>       jnz    short get_env_string_seq_number_next
4887                            <1>
4888 0000A729 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size ; +512 (+4096)
4889 0000A72F F5                <1>       cmc
4890 0000A730 72EF              <1>       jc     short get_env_string_stc_retn
4891                            <1>
4892 0000A732 AC                <1>       lodsb
4893 0000A733 3C01              <1>       cmp    al, 1
4894 0000A735 72EA              <1>       jb     short get_env_string_stc_retn
4895 0000A737 FEC1              <1>       inc    cl
4896 0000A739 EBBF              <1>       jmp    short get_env_string_seq_number_check
4897                            <1>
4898                            <1> get_env_string_with_word:
4899 0000A73B 31C9              <1>       xor    ecx, ecx
4900                            <1>
4901                            <1> get_env_string_calc_word_length:
4902 0000A73D AC                <1>       lodsb
4903 0000A73E 3C20              <1>       cmp    al, 20h
4904 0000A740 7211              <1>       jb     short get_env_string_calc_word_length_ok
4905                            <1>       ;inc   cx
4906 0000A742 FEC1              <1>       inc    cl
4907                            <1>
4908 0000A744 3C61              <1>       cmp    al, 'a'
4909 0000A746 72F5              <1>       jb     short get_env_string_calc_word_length
4910 0000A748 3C7A              <1>       cmp    al, 'z'
4911 0000A74A 77F1              <1>       ja     short get_env_string_calc_word_length
4912 0000A74C 24DF              <1>       and    al, 0DFh
4913 0000A74E 8846FF            <1>       mov    [esi-1], al
4914 0000A751 EBEA              <1>       jmp    short get_env_string_calc_word_length
4915                            <1>
4916                            <1> get_env_string_calc_word_length_ok:
4917 0000A753 08C9              <1>       or     cl, cl
4918 0000A755 7506              <1>       jnz    short get_env_string_calc_word_length_save
4919                            <1>
4920 0000A757 5E                <1>       pop    esi ; **
4921                            <1>
4922                            <1> get_env_string_stc_retn1:
4923 0000A758 59                <1>       pop    ecx ; *
4924                            <1>
4925                            <1> get_env_string_with_word_stc_retn:
4926 0000A759 31C0              <1>       xor    eax, eax
4927 0000A75B F9                <1>       stc
4928 0000A75C C3                <1>       retn
4929                            <1>
4930                            <1> get_env_string_calc_word_length_save:
4931 0000A75D 871C24            <1>       xchg   ebx, [esp] ; **
4932 0000A760 89DE              <1>       mov    esi, ebx
4933                            <1>              ; Start of the env string (to be searched)
4934                            <1>
4935 0000A762 57                <1>       push   edi ; ***
4936 0000A763 89D7              <1>       mov    edi, edx ; Env_Page
4937                            <1>
4938                            <1> get_env_string_compare:
4939 0000A765 57                <1>       push   edi ; ****
4940 0000A766 51                <1>       push   ecx ; ***** ; Variable name length
4941                            <1>
4942                            <1> get_env_string_compare_rep:
4943 0000A767 AC                <1>       lodsb
4944 0000A768 AE                <1>       scasb
4945 0000A769 7511              <1>       jne    short get_env_string_compare_next1
4946 0000A76B E2FA              <1>       loop   get_env_string_compare_rep
4947                            <1>
4948 0000A76D 803F3D            <1>       cmp    byte [edi], '='
4949 0000A770 750A              <1>       jne    short get_env_string_compare_next1
4950                            <1>
4951 0000A772 59                <1>       pop    ecx ; *****
4952 0000A773 5F                <1>       pop    edi ; ****
4953 0000A774 89FE              <1>       mov    esi, edi
4954 0000A776 5F                <1>       pop    edi ; ***
4955 0000A777 871C24            <1>       xchg   ebx, [esp] ; **
4956 0000A77A EB82              <1>       jmp    short get_env_string_move_to_buff
4957                            <1>
4958                            <1> get_env_string_compare_next1:
4959 0000A77C 89FE              <1>       mov    esi, edi
4960 0000A77E 59                <1>       pop    ecx ; *****
4961 0000A77F 5F                <1>       pop    edi ; ****
4962                            <1> get_env_string_compare_next2:
4963 0000A780 81FEFF310900      <1>       cmp    esi, Env_Page + Env_Page_Size - 1 ; +511 (+4095)
4964 0000A786 7310              <1>       jnb    short get_env_string_compare_not_ok
4965 0000A788 20C0              <1>       and    al, al
4966 0000A78A AC                <1>       lodsb
4967 0000A78B 75F3              <1>       jnz    short get_env_string_compare_next2
4968 0000A78D 08C0              <1>       or     al, al
4969 0000A78F 7407              <1>       jz     short get_env_string_compare_not_ok
4970 0000A791 4E                <1>       dec    esi ; 12/04/2016
4971 0000A792 89F7              <1>       mov    edi, esi
4972 0000A794 89DE              <1>       mov    esi, ebx
4973 0000A796 EBCD              <1>       jmp    short get_env_string_compare
4974                            <1>
4975                            <1> get_env_string_compare_not_ok:
4976 0000A798 5F                <1>       pop    edi ; ***
4977 0000A799 89DE              <1>       mov    esi, ebx
4978 0000A79B 5B                <1>       pop    ebx ; **
```

```
4979 0000A79C EBBA              <1>       jmp    short get_env_string_stc_retn1
4980                            <1>
4981                            <1> set_environment_string:
4982                            <1>       ; 13/04/2016
4983                            <1>       ; 12/04/2016
4984                            <1>       ; 11/04/2016
4985                            <1>       ; 06/04/2016
4986                            <1>       ; 05/04/2016 (TRDOS 386 = TRDOS v2.0)
4987                            <1>       ; 02/09/2011 (TRDOS v1, MAINPROG.ASM)
4988                            <1>       ; 29/08/2011
4989                            <1>       ; 29/08/2011
4990                            <1>       ; INPUT->
4991                            <1>       ;      ESI = ASCIIZ environment string
4992                            <1>       ; OUTPUT ->
4993                            <1>       ;      ESI is not changed
4994                            <1>       ;      CF = 1 -> Could not set,
4995                            <1>       ;           insufficient environment space
4996                            <1>       ;
4997                            <1>       ; (EAX, EDX will be changed)
4998                            <1>       ;
4999                            <1>       ;    (EAX = Start address of the env string if > 0)
5000                            <1>       ;    (EDX = Environment string length)
5001                            <1>
5002 0000A79E 56                <1>       push   esi ; *
5003                            <1>
5004 0000A79F 31C0              <1>       xor    eax, eax
5005                            <1>
5006                            <1> set_env_chk_validation1:
5007 0000A7A1 FEC4              <1>       inc    ah ; variable (string) length
5008 0000A7A3 AC                <1>       lodsb
5009 0000A7A4 3C3D              <1>       cmp    al, '='
5010 0000A7A6 7415              <1>       je     short set_env_chk_validation2
5011 0000A7A8 3C20              <1>       cmp    al, 20h
5012 0000A7AA 720F              <1>       jb     short set_env_string_stc
5013                            <1>
5014                            <1>       ; 06/04/2016
5015 0000A7AC 3C61              <1>       cmp    al, 'a'
5016 0000A7AE 72F1              <1>       jb     short set_env_chk_validation1
5017 0000A7B0 3C7A              <1>       cmp    al, 'z'
5018 0000A7B2 77ED              <1>       ja     short set_env_chk_validation1
5019 0000A7B4 2C20              <1>       sub    al, 'a'-'A'
5020 0000A7B6 8846FF            <1>       mov    [esi-1], al
5021 0000A7B9 EBE6              <1>       jmp    short set_env_chk_validation1
5022                            <1>
5023                            <1> set_env_string_stc:
5024 0000A7BB 5E                <1>       pop    esi ; *
5025                            <1>       ;stc
5026 0000A7BC C3                <1>       retn
5027                            <1>
5028                            <1> set_env_chk_validation2:
5029 0000A7BD 51                <1>       push   ecx ; **
5030 0000A7BE 53                <1>       push   ebx ; ***
5031 0000A7BF 57                <1>       push   edi ; ****
5032                            <1>
5033                            <1>       ; 12/04/2016
5034 0000A7C0 8B5C240C          <1>       mov    ebx, [esp+12]
5035                            <1>
5036                            <1> set_env_chk_validation2w:
5037 0000A7C4 89F7              <1>       mov    edi, esi
5038 0000A7C6 4F                <1>       dec    edi
5039                            <1>
5040 0000A7C7 807FFF20          <1>       cmp    byte [edi-1], 20h
5041 0000A7CB 771A              <1>       ja     short set_env_chk_validation2z
5042                            <1>
5043 0000A7CD 56                <1>       push   esi
5044 0000A7CE 89FE              <1>       mov    esi, edi
5045 0000A7D0 4E                <1>       dec    esi
5046                            <1>
5047                            <1> set_env_chk_validation2x:
5048 0000A7D1 4E                <1>       dec    esi
5049                            <1>
5050 0000A7D2 39DE              <1>       cmp    esi, ebx
5051 0000A7D4 7207              <1>       jb     short set_env_chk_validation2y
5052                            <1>
5053 0000A7D6 4F                <1>       dec    edi
5054                            <1>
5055 0000A7D7 8A06              <1>       mov    al, [esi]
5056 0000A7D9 8807              <1>       mov    [edi], al
5057                            <1>
5058 0000A7DB EBF4              <1>       jmp    short set_env_chk_validation2x
5059                            <1>
5060                            <1> set_env_chk_validation2y:
5061 0000A7DD 5E                <1>       pop    esi
5062                            <1>
5063                            <1>       ;mov   byte [ebx], 20h
5064                            <1>
5065 0000A7DE 43                <1>       inc    ebx
5066 0000A7DF 895C240C          <1>       mov    [esp+12], ebx
5067                            <1>
5068 0000A7E3 FECC              <1>       dec    ah ; 13/04/2016
5069                            <1>
5070 0000A7E5 EBDD              <1>       jmp    short set_env_chk_validation2w
5071                            <1>
5072                            <1> set_env_chk_validation2z:
5073 0000A7E7 BA00300900        <1>       mov    edx, Env_Page
5074 0000A7EC 89D7              <1>       mov    edi, edx
5075                            <1>
5076                            <1> set_env_chk_validation3:
5077 0000A7EE AC                <1>       lodsb
5078 0000A7EF 3C20              <1>       cmp    al, 20h
5079 0000A7F1 74FB              <1>       je     short set_env_chk_validation3
5080                            <1>
5081 0000A7F3 9C                <1>       pushf
5082                            <1>
5083                            <1>       ; 12/04/2016
```

```
                                <1> set_env_chk_validation3n:
0000A7F4 3C61                   <1>       cmp    al, 'a'
0000A7F6 720C                   <1>       jb     short set_env_chk_validation3c
0000A7F8 3C7A                   <1>       cmp    al, 'z'
0000A7FA 7705                   <1>       ja     short set_env_chk_validation3x
0000A7FC 2C20                   <1>       sub    al, 'a'-'A'
0000A7FE 8846FF                 <1>       mov    [esi-1], al
                                <1>
                                <1> set_env_chk_validation3x:
0000A801 AC                     <1>       lodsb
0000A802 EBF0                   <1>       jmp    short set_env_chk_validation3n
                                <1>
                                <1> set_env_chk_validation3c:
0000A804 3C20                   <1>       cmp    al, 20h
0000A806 73F9                   <1>       jnb    short set_env_chk_validation3x
                                <1>
0000A808 803F00                 <1>       cmp    byte [edi], 0
0000A80B 7731                   <1>       ja     short set_env_chk_validation4
                                <1>
0000A80D 9D                     <1>       popf
0000A80E 7228                   <1>       jb     short set_env_string_nothing
                                <1>
0000A810 B900020000             <1>       mov    ecx, Env_Page_Size ; 512 (4096)
                                <1>
0000A815 89DE                   <1>       mov    esi, ebx ; 12/04/2016
                                <1>
                                <1> set_env_string_copy_to_envb:
0000A817 AC                     <1>       lodsb
0000A818 3C20                   <1>       cmp    al, 20h
0000A81A 720A                   <1>       jb     short set_env_string_copy_to_envb_z
0000A81C AA                     <1>       stosb
0000A81D E2F8                   <1>       loop   set_env_string_copy_to_envb
                                <1>
                                <1>       ; 11/04/2016
0000A81F 89D7                   <1>       mov    edi, edx ; Env_Page
0000A821 B900020000             <1>       mov    ecx, Env_Page_Size
                                <1>
                                <1> set_env_string_copy_to_envb_z:
0000A826 52                     <1>       push   edx  ; Start address of the variable
0000A827 BA00020000             <1>       mov    edx, Env_Page_Size
0000A82C 29CA                   <1>       sub    edx, ecx ; variable (string) length
                                <1>
0000A82E 28C0                   <1>       sub    al, al ; 0
0000A830 F3AA                   <1>       rep    stosb ; clear remain bytes of the env page
                                <1>
0000A832 58                     <1>       pop    eax  ; Start address of the variable
                                <1>
                                <1> set_env_string_allocate_envb_retn:  ; stc or clc return
0000A833 5F                     <1>       pop    edi ; ****
0000A834 5B                     <1>       pop    ebx ; ***
0000A835 59                     <1>       pop    ecx ; **
0000A836 5E                     <1>       pop    esi ; *
0000A837 C3                     <1>       retn
                                <1>
                                <1> set_env_string_nothing:
0000A838 31C0                   <1>       xor    eax, eax
0000A83A 31D2                   <1>       xor    edx, edx ; 11/04/2016
0000A83C EBF5                   <1>       jmp    short set_env_string_allocate_envb_retn
                                <1>
                                <1> set_env_chk_validation4:
                                <1>       ; 11/04/2016
0000A83E 9D                     <1>       popf
                                <1>
0000A83F 89D6                   <1>       mov    esi, edx  ; Env_Page
                                <1>
                                <1> set_env_chk_validation5:
0000A841 89DF                   <1>       mov    edi, ebx  ; ASCIIZ environment string address
0000A843 0FB6CC                 <1>       movzx  ecx, ah ; Variable (string) length (with '=')
                                <1>
                                <1> set_env_chk_validation5_loop:
0000A846 AC                     <1>       lodsb
0000A847 AE                     <1>       scasb
0000A848 750A                   <1>       jne    short set_env_chk_validation6
0000A84A E2FA                   <1>       loop   set_env_chk_validation5_loop
                                <1>
0000A84C 3C3D                   <1>       cmp    al, '='
0000A84E 0F8483000000           <1>         je      set_env_change_variable
                                <1>
                                <1> set_env_chk_validation6:
0000A854 08C0                   <1>       or     al, al ; 0
0000A856 7403                   <1>       jz     short set_env_chk_validation7
                                <1>
0000A858 AC                     <1>       lodsb
0000A859 EBF9                   <1>       jmp    short set_env_chk_validation6
                                <1>
                                <1> set_env_chk_validation7:
0000A85B 88E1                   <1>       mov    cl, ah
0000A85D 01F1                   <1>       add    ecx, esi
0000A85F 81F9FF310900           <1>       cmp    ecx, Env_Page + Env_Page_Size - 1
                                <1>              ; 511 (4095)
                                <1>              ; strlen + '=' + 0
0000A865 72DA                   <1>       jb     short set_env_chk_validation5
                                <1>
                                <1> set_env_chk_validation8: ; variable not found
0000A867 0FB6F4                 <1>       movzx  esi, ah  ; variable name length (with '=')
0000A86A 01DE                   <1>       add    esi, ebx ; position just after of the '='
                                <1>
                                <1> set_env_chk_validation8_loop:
0000A86C AC                     <1>       lodsb
0000A86D 3C20                   <1>       cmp    al, 20h
0000A86F 74FB                   <1>       je     short set_env_chk_validation8_loop
0000A871 72C5                   <1>       jb     short set_env_string_nothing
                                <1>
                                <1> set_env_chk_validation9:
0000A873 AC                     <1>       lodsb
```

```
5189 0000A874 3C20              <1>       cmp    al, 20h
5190 0000A876 73FB              <1>       jnb    short set_env_chk_validation9
5191                            <1>
5192                            <1>       ; End of ASCIIZ environment string
5193                            <1>
5194                            <1> set_env_add_variable:
5195 0000A878 29DE              <1>       sub    esi, ebx ; variable+definition length
5196                            <1>
5197 0000A87A 56                <1>       push   esi ; *****
5198                            <1>
5199 0000A87B 89D6              <1>       mov    esi, edx ; Environment page address
5200                            <1>
5201 0000A87D B900020000        <1>       mov    ecx, Env_Page_Size ; 512 (4096)
5202                            <1>
5203                            <1> set_env_add_variable_loop:
5204 0000A882 AC                <1>       lodsb
5205 0000A883 20C0              <1>       and    al, al
5206 0000A885 7406              <1>       jz     short set_env_add_variable_chk1 ; 0
5207 0000A887 E2F9              <1>       loop   set_env_add_variable_loop
5208                            <1>
5209                            <1>       ; 11/04/2016
5210 0000A889 884EFF            <1>       mov    [esi-1], cl ; 0
5211 0000A88C 41                <1>       inc    ecx
5212                            <1>
5213                            <1> set_env_add_variable_chk1:
5214 0000A88D 49                <1>       dec    ecx
5215 0000A88E 7408              <1>       jz     short set_env_add_variable_nspc
5216 0000A890 AC                <1>       lodsb
5217 0000A891 08C0              <1>       or     al, al
5218 0000A893 740C              <1>       jz     short set_env_add_variable_chk2 ; 00
5219 0000A895 49                <1>       dec    ecx
5220 0000A896 75EA              <1>       jnz    short set_env_add_variable_loop
5221                            <1>
5222                            <1> set_env_add_variable_nspc: ; no space on environment page
5223 0000A898 58                <1>       pop    eax ; *****
5224 0000A899 B808000000        <1>       mov    eax, 8 ; No space for new environment string
5225 0000A89E F9                <1>       stc
5226 0000A89F EB92              <1>         jmp     short set_env_string_allocate_envb_retn
5227                            <1>
5228                            <1> set_env_add_variable_chk2:
5229 0000A8A1 8B0C24            <1>       mov    ecx, [esp] ; *****
5230 0000A8A4 4E                <1>       dec    esi ; beginning address of the new variable
5231 0000A8A5 89F0              <1>       mov    eax, esi
5232 0000A8A7 01C8              <1>       add    eax, ecx ; string length (with CR)
5233 0000A8A9 81C200020000      <1>       add    edx, Env_Page_Size ; 512 (4096)
5234 0000A8AF 39D0              <1>       cmp    eax, edx
5235 0000A8B1 77E5              <1>       ja     short set_env_add_variable_nspc
5236 0000A8B3 49                <1>       dec    ecx ; except CR at the end
5237 0000A8B4 89CA              <1>       mov    edx, ecx ; 12/04/2016
5238 0000A8B6 89F7              <1>       mov    edi, esi
5239 0000A8B8 893C24            <1>       mov    [esp], edi ; ***** ; Start address of new variable
5240 0000A8BB 89DE              <1>       mov    esi, ebx ; ASCIIZ environment string address
5241 0000A8BD F3A4              <1>       rep    movsb
5242 0000A8BF 28C0              <1>       sub    al, al
5243 0000A8C1 AA                <1>       stosb
5244 0000A8C2 58                <1>       pop    eax ; ***** ; Beginning address of new variable
5245 0000A8C3 81FF00320900      <1>         cmp     edi, Env_Page + Env_Page_Size ; 12/04/2016
5246 0000A8C9 0F8364FFFFFF      <1>         jnb     set_env_string_allocate_envb_retn ; OK !
5247 0000A8CF 880F              <1>       mov    [edi], cl ; 0
5248 0000A8D1 F8                <1>       clc    ; 13/04/2016
5249 0000A8D2 E95CFFFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
5250                            <1>
5251                            <1> set_env_change_variable:
5252                            <1>       ; 06/04/2016
5253                            <1>       ; esi = Variable's address in environment page (after '=')
5254                            <1>       ; edi = ASCIIZ environment string address (after '=')
5255                            <1>
5256                            <1>       ; ah = variable length from start to the '='
5257 0000A8D7 8825[48960100]    <1>       mov    [env_var_length], ah
5258                            <1>
5259 0000A8DD 28C9              <1>       sub    cl, cl ; ecx = 0
5260                            <1>
5261 0000A8DF 57                <1>       push   edi ; *****
5262                            <1>
5263 0000A8E0 89F7              <1>       mov    edi, esi ; 11/04/2016
5264                            <1>
5265                            <1> set_env_change_variable_calc1:
5266 0000A8E2 AC                <1>       lodsb
5267 0000A8E3 08C0              <1>       or     al, al
5268 0000A8E5 7403              <1>       jz     short set_env_change_variable_calc2
5269                            <1>
5270 0000A8E7 41                <1>       inc    ecx ; length of environment string (after the '=')
5271                            <1>
5272 0000A8E8 EBF8              <1>       jmp    short set_env_change_variable_calc1
5273                            <1>
5274                            <1> set_env_change_variable_calc2:
5275 0000A8EA 8B3424            <1>       mov    esi, [esp] ; ASCIIZ environment string address
5276                            <1>
5277 0000A8ED 29D2              <1>       sub    edx, edx
5278                            <1>
5279                            <1> set_env_change_variable_calc3:
5280 0000A8EF AC                <1>       lodsb
5281 0000A8F0 3C20              <1>       cmp    al, 20h
5282 0000A8F2 7203              <1>       jb     short set_env_change_variable_calc4
5283                            <1>
5284 0000A8F4 42                <1>       inc    edx ; length of ASCIIZ string (after the '=')
5285                            <1>
5286 0000A8F5 EBF8              <1>       jmp    short set_env_change_variable_calc3
5287                            <1>
5288                            <1> set_env_change_variable_calc4:
5289 0000A8F7 C646FF00          <1>       mov    byte [esi-1], 0  ; put ZERO instead of CR
5290                            <1>
5291 0000A8FB 5E                <1>       pop    esi ; ***** ; ASCIIZ string address (after '=')
5292                            <1>
5293                            <1>       ; EDI = Old variable's address (after '=')
```

```
                                <1>
                                <1>       ; compare the new string with the old string
0000A8FC 39CA                   <1>       cmp    edx, ecx
0000A8FE 7717                   <1>       ja     short set_env_change_variable_calc5 ; longer
0000A900 0F828F000000           <1>         jb      set_env_change_variable_calc9 ; shorter
                                <1>
                                <1>       ;same length (simple copy)
0000A906 0FB6C4                 <1>       movzx  eax, ah
0000A909 01C2                   <1>       add    edx, eax
0000A90B F7D8                   <1>       neg    eax
0000A90D 01F8                   <1>       add    eax, edi
                                <1>       ; EAX = Start address of the variable
                                <1>       ; EDX = Variable length (without ZERO at the end of variable)
                                <1>
0000A90F F3A4                   <1>       rep    movsb
0000A911 F8                     <1>       clc    ; 13/04/2016
0000A912 E91CFFFFFF             <1>         jmp     set_env_string_allocate_envb_retn ; OK !
                                <1>
                                <1> set_env_change_variable_calc5:
                                <1>       ; 11/04/2016
0000A917 52                     <1>       push   edx ; *****
0000A918 29CA                   <1>       sub    edx, ecx ; difference ; (the new string is longer)
0000A91A 89F3                   <1>       mov    ebx, esi
0000A91C 89FE                   <1>       mov    esi, edi
                                <1>
                                <1> set_env_change_variable_calc6:
0000A91E AC                     <1>       lodsb
0000A91F 20C0                   <1>       and    al, al
0000A921 75FB                   <1>       jnz    short set_env_change_variable_calc6
                                <1>
0000A923 81FE00320900           <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
0000A929 0F8369FFFFFF           <1>         jnb     set_env_add_variable_nspc
                                <1>
0000A92F 89F9                   <1>       mov    ecx, edi  ; current (old) variable's address
0000A931 89F7                   <1>       mov    edi, esi  ; next variable's address
                                <1>
0000A933 AC                     <1>       lodsb
0000A934 08C0                   <1>       or     al, al
0000A936 7416                   <1>       jz     short set_env_change_variable_calc8 ; 00
                                <1>
                                <1> set_env_change_variable_calc7:
0000A938 AC                     <1>       lodsb
0000A939 20C0                   <1>       and    al, al
0000A93B 75FB                   <1>       jnz    short set_env_change_variable_calc7
                                <1>
0000A93D 81FE00320900           <1>       cmp    esi, Env_Page + Env_Page_Size ; 512 (4096)
0000A943 0F834FFFFFFF           <1>         jnb     set_env_add_variable_nspc
                                <1>
0000A949 AC                     <1>       lodsb
0000A94A 08C0                   <1>       or     al, al
0000A94C 75EA                   <1>       jnz    short set_env_change_variable_calc7
                                <1>
                                <1> set_env_change_variable_calc8:
0000A94E 4E                     <1>       dec    esi ; address of the second (last) 0 of the 00
                                <1>
0000A94F 01F2                   <1>       add    edx, esi ; final position of the last 0
                                <1>
0000A951 81FA00320900           <1>       cmp    edx, Env_Page + Env_Page_Size ; 512 (4096)
0000A957 0F833BFFFFFF           <1>         jnb     set_env_add_variable_nspc
                                <1>
0000A95D 89C8                   <1>       mov    eax, ecx ; old variable's address (after '=')
                                <1>
0000A95F 89F1                   <1>       mov    ecx, esi
0000A961 29F9                   <1>       sub    ecx, edi ; count of bytes to move forward
                                <1>
                                <1>       ; 13/04/2016
0000A963 C60200                 <1>       mov    byte [edx], 0
0000A966 89D7                   <1>       mov    edi, edx
0000A968 29F2                   <1>       sub    edx, esi ; difference (additional byte count)
0000A96A 4F                     <1>       dec    edi ; the last zero address (first byte of the 00)
0000A96B 89FE                   <1>       mov    esi, edi
0000A96D 29D6                   <1>       sub    esi, edx ; - displacement
                                <1>
0000A96F FA                     <1>       cli    ; disable interrupts
0000A970 FD                     <1>       std    ; backward
                                <1>
0000A971 F3A4                   <1>       rep    movsb ; move ECX bytes from DS:ESI to ES:EDI
                                <1>
0000A973 FC                     <1>       cld    ; forward (default)
0000A974 FB                     <1>       sti    ; enable interrupts
                                <1>
0000A975 89C7                   <1>       mov    edi, eax
0000A977 59                     <1>       pop    ecx ; ***** ; byte count (after '=')
0000A978 89CA                   <1>       mov    edx, ecx
0000A97A 89DE                   <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
0000A97C 89FB                   <1>       mov    ebx, edi
                                <1>
0000A97E F3A4                   <1>       rep    movsb
                                <1>
0000A980 880F                   <1>       mov    [edi], cl ; 0 ; end of variable
                                <1>
0000A982 0FB605[48960100]       <1>       movzx  eax, byte [env_var_length]
0000A989 01C2                   <1>       add    edx, eax ; variable length (total)
0000A98B F7D8                   <1>       neg    eax
0000A98D 01D8                   <1>       add    eax, ebx ; start address of the variable
0000A98F F8                     <1>       clc    ; 13/04/2016
0000A990 E99EFEFFFF             <1>         jmp     set_env_string_allocate_envb_retn ; OK !
                                <1>
                                <1> set_env_change_variable_calc9:
                                <1>       ; 11/04/2016
0000A995 21D2                   <1>       and    edx, edx ; is empty ?
0000A997 753B                   <1>       jnz    short set_env_change_variable_calc15
                                <1>
0000A999 0FB6DC                 <1>       movzx  ebx, ah
0000A99C F7DB                   <1>       neg    ebx
```

```
5399 0000A99E 01FB              <1>       add    ebx, edi
5400                            <1>
5401                            <1>       ; EBX = Start address of the variable (in env page)
5402                            <1>       ; EDX = Variable length = 0
5403                            <1>
5404 0000A9A0 89FE              <1>       mov    esi, edi
5405                            <1>
5406                            <1> set_env_change_variable_calc10:
5407 0000A9A2 AC                <1>       lodsb
5408 0000A9A3 08C0              <1>       or     al, al
5409 0000A9A5 75FB              <1>       jnz    short set_env_change_variable_calc10
5410                            <1>
5411 0000A9A7 B9FF310900        <1>       mov    ecx, Env_Page + Env_Page_Size - 1
5412                            <1>
5413 0000A9AC 39CE              <1>       cmp    esi, ecx ; +511 (+4095)
5414 0000A9AE 7604              <1>       jna    short set_env_change_variable_calc11
5415                            <1>
5416 0000A9B0 89CE              <1>       mov    esi, ecx
5417 0000A9B2 8806              <1>       mov    [esi], al ; 0
5418                            <1>
5419                            <1> set_env_change_variable_calc11:
5420 0000A9B4 89DF              <1>       mov    edi, ebx ; old variable's start address
5421                            <1>
5422                            <1> set_env_change_variable_calc12:
5423 0000A9B6 AC                <1>       lodsb
5424 0000A9B7 AA                <1>       stosb
5425 0000A9B8 20C0              <1>       and    al, al
5426 0000A9BA 75FA              <1>       jnz    short set_env_change_variable_calc12
5427 0000A9BC 39CE              <1>       cmp    esi, ecx
5428 0000A9BE 7706              <1>       ja     short set_env_change_variable_calc13
5429 0000A9C0 AC                <1>       lodsb
5430 0000A9C1 AA                <1>       stosb
5431 0000A9C2 20C0              <1>       and    al, al
5432 0000A9C4 75F0              <1>       jnz    short set_env_change_variable_calc12
5433                            <1>
5434                            <1> set_env_change_variable_calc13:
5435 0000A9C6 29F9              <1>       sub    ecx, edi
5436 0000A9C8 7203              <1>       jb     short set_env_change_variable_calc14
5437 0000A9CA 41                <1>       inc    ecx ; 1-512 (1-4096)
5438 0000A9CB F3AA              <1>       rep    stosb ; al = 0
5439                            <1>
5440                            <1> set_env_change_variable_calc14:
5441 0000A9CD 29C0              <1>       sub    eax, eax ; Start address of the variable
5442                            <1>       ; EAX = 0 -> Variable is removed
5443                            <1>       ; EDX = Variable length = 0
5444                            <1>
5445 0000A9CF E95FFEFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
5446                            <1>
5447                            <1> set_env_change_variable_calc15:
5448 0000A9D4 52                <1>       push   edx ; *****
5449 0000A9D5 F7DA              <1>       neg    edx
5450 0000A9D7 01CA              <1>       add    edx, ecx ; difference (the old string is longer)
5451 0000A9D9 89F3              <1>       mov    ebx, esi
5452 0000A9DB 89FE              <1>       mov    esi, edi
5453                            <1>
5454                            <1> set_env_change_variable_calc16:
5455 0000A9DD AC                <1>       lodsb
5456 0000A9DE 20C0              <1>       and    al, al
5457 0000A9E0 75FB              <1>       jnz    short set_env_change_variable_calc16
5458                            <1>
5459 0000A9E2 B900320900        <1>       mov    ecx, Env_Page + Env_Page_Size
5460                            <1>
5461 0000A9E7 39CE              <1>       cmp    esi, ecx ; +512 (+4096)
5462 0000A9E9 7605              <1>       jna    short set_env_change_variable_calc17
5463                            <1>
5464 0000A9EB 89CE              <1>       mov    esi, ecx
5465 0000A9ED 8846FF            <1>       mov    [esi-1], al ; 0
5466                            <1>
5467                            <1> set_env_change_variable_calc17:
5468 0000A9F0 89F9              <1>       mov    ecx, edi  ; current (old) variable's address
5469 0000A9F2 89F7              <1>       mov    edi, esi  ; next variable's address
5470                            <1>
5471 0000A9F4 AC                <1>       lodsb
5472 0000A9F5 08C0              <1>       or     al, al
5473 0000A9F7 741D              <1>       jz     short set_env_change_variable_calc20
5474                            <1>
5475                            <1> set_env_change_variable_calc18:
5476 0000A9F9 AC                <1>       lodsb
5477 0000A9FA 20C0              <1>       and    al, al
5478 0000A9FC 75FB              <1>       jnz    short set_env_change_variable_calc18
5479                            <1>
5480 0000A9FE 81FE00320900      <1>       cmp    esi, Env_Page + Env_Page_Size
5481 0000AA04 720B              <1>       jb     short set_env_change_variable_calc19
5482 0000AA06 740E              <1>       je     short set_env_change_variable_calc20
5483                            <1>
5484 0000AA08 BEFF310900        <1>       mov    esi, Env_Page + Env_Page_Size - 1
5485 0000AA0D 8806              <1>       mov    [esi], al ; 0
5486 0000AA0F EB06              <1>       jmp    short set_env_change_variable_calc21
5487                            <1>
5488                            <1> set_env_change_variable_calc19:
5489 0000AA11 AC                <1>       lodsb
5490 0000AA12 08C0              <1>       or     al, al
5491 0000AA14 75E3              <1>       jnz    short set_env_change_variable_calc18
5492                            <1>
5493                            <1> set_env_change_variable_calc20:
5494 0000AA16 4E                <1>       dec    esi ; address of the second (last) 0 of the 00
5495                            <1>
5496                            <1> set_env_change_variable_calc21:
5497                            <1>       ; edx = difference (byte count)
5498                            <1>
5499 0000AA17 89C8              <1>       mov    eax, ecx ; old variable's address (after '=')
5500                            <1>
5501 0000AA19 89F1              <1>       mov    ecx, esi
5502 0000AA1B 29F9              <1>       sub    ecx, edi ; count of bytes to move backward
5503                            <1>
```

```
5504 0000AA1D 89FE              <1>       mov    esi, edi ; next variable's address
5505 0000AA1F 29D7              <1>       sub    edi, edx ; (displacement)
5506                            <1>
5507 0000AA21 F3A4              <1>       rep    movsb
5508                            <1>
5509 0000AA23 880F              <1>       mov    [edi], cl ; 0 ; 00 ; end of environment variables
5510                            <1>
5511 0000AA25 89C7              <1>       mov    edi, eax
5512 0000AA27 5A                <1>       pop    edx ; ***** ; byte count (after '=')
5513 0000AA28 89D1              <1>       mov    ecx, edx
5514 0000AA2A 89DE              <1>       mov    esi, ebx ; ASCIIZ string address (after '=')
5515 0000AA2C 89FB              <1>       mov    ebx, edi
5516                            <1>
5517 0000AA2E F3A4              <1>       rep    movsb
5518                            <1>
5519 0000AA30 880F              <1>       mov    [edi], cl ; 0 ; end of variable
5520                            <1>
5521 0000AA32 0FB605[48960100]  <1>       movzx  eax, byte [env_var_length]
5522 0000AA39 01C2              <1>       add    edx, eax ; variable length (total)
5523 0000AA3B F7D8              <1>       neg    eax
5524 0000AA3D 01D8              <1>       add    eax, ebx ; start address of the variable
5525 0000AA3F F8                <1>       clc    ; 13/04/2016
5526 0000AA40 E9EEFDFFFF        <1>         jmp     set_env_string_allocate_envb_retn ; OK !
5527                            <1>
5528                            <1> mainprog_startup_configuration:
5529                            <1>       ; 22/11/2017
5530                            <1>       ; 06/05/2016
5531                            <1>       ; 14/04/2016 (TRDOS 386 = TRDOS v2.0)
5532                            <1>       ; 17/09/2011 (TRDOS v1, MAINPROG.ASM)
5533                            <1>       ;
5534                            <1> loc_load_mainprog_cfg_file:
5535 0000AA45 BE[EC400100]      <1>       mov    esi, MainProgCfgFile
5536 0000AA4A 66B80018          <1>       mov    ax, 1800h ; Except volume label and dirs
5537 0000AA4E E83EEAFFFF        <1>       call   find_first_file
5538 0000AA53 7256              <1>       jc     short loc_load_mainprog_cfg_exit
5539                            <1>
5540                            <1>       ;or    eax, eax
5541                            <1>       ;jz    short loc_load_mainprog_cfg_exit
5542                            <1>
5543                            <1> loc_start_mainprog_configuration:
5544                            <1>       ; ESI = FindFile_DirEntry Location
5545                            <1>       ; EAX = File Size
5546                            <1>
5547 0000AA55 A3[CC8A0100]      <1>       mov    [MainProgCfg_FileSize], eax
5548                            <1>
5549 0000AA5A 668B5614          <1>       mov    dx, [esi+DirEntry_FstClusHI]
5550 0000AA5E C1E210            <1>       shl    edx, 16
5551 0000AA61 668B561A          <1>       mov    dx, [esi+DirEntry_FstClusLO]
5552 0000AA65 8915[FC950100]    <1>       mov    [csftdf_sf_cluster], edx
5553                            <1>
5554 0000AA6B 89C1              <1>       mov    ecx, eax
5555 0000AA6D 29C0              <1>       sub    eax, eax
5556                            <1>
5557                            <1>       ; TRDOS 386 (TRDOS v2.0)
5558                            <1>       ; Allocate contiguous memory block for loading the file
5559                            <1>
5560                            <1>       ; eax = 0 (Allocate memory from the beginning)
5561                            <1>       ; ecx = File (Allocation) size in bytes
5562                            <1>
5563 0000AA6F E811BAFFFF        <1>       call   allocate_memory_block
5564 0000AA74 7235              <1>       jc     short loc_load_mainprog_cfg_exit
5565                            <1>
5566 0000AA76 A3[F4950100]      <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
5567 0000AA7B 890D[F8950100]    <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
5568                            <1>
5569 0000AA81 31DB              <1>       xor    ebx, ebx
5570                            <1>       ;mov   [csftdf_sf_rbytes], ebx ; 0, reset
5571                            <1>
5572 0000AA83 8A3D[DE8A0100]    <1>       mov    bh, [Current_Drv] ; [FindFile_Drv]
5573 0000AA89 BE00010900        <1>       mov    esi, Logical_DOSDisks
5574 0000AA8E 01DE              <1>       add    esi, ebx
5575                            <1>
5576 0000AA90 8B1D[F4950100]    <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
5577                            <1>
5578 0000AA96 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
5579 0000AA9A 7710              <1>         ja   short loc_mcfg_load_fat_file
5580                            <1>
5581 0000AA9C C705[04960100]0000- <1>     mov    dword [csftdf_r_size], 65536
5581 0000AAA4 0100              <1>
5582 0000AAA6 E9A1010000        <1>         jmp     loc_mcfg_load_fs_file
5583                            <1>
5584                            <1> loc_load_mainprog_cfg_exit:
5585 0000AAAB C3                <1>       retn
5586                            <1>
5587                            <1> loc_mcfg_load_fat_file:
5588 0000AAAC 0FB74611          <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
5589 0000AAB0 0FB64E13          <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
5590 0000AAB4 F7E1              <1>       mul    ecx
5591 0000AAB6 A3[04960100]      <1>       mov    [csftdf_r_size], eax
5592                            <1>
5593                            <1> loc_mcfg_load_fat_file_next:
5594 0000AABB E822010000        <1>       call   mcfg_read_fat_file_sectors
5595 0000AAC0 0F8206010000      <1>         jc      mcfg_deallocate_mem
5596                            <1>
5597 0000AAC6 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
5598 0000AAC8 74F1              <1>       jz     short loc_mcfg_load_fat_file_next
5599                            <1>
5600                            <1> loc_mcfg_load_fat_file_ok:
5601                            <1>       ; 06/05/2016
5602 0000AACA C705[98960100]-   <1>       mov    dword [mainprog_return_addr], loc_mcfg_ci_return_addr
5602 0000AAD0 [8DAB0000]        <1>
5603                            <1>       ;
5604 0000AAD4 8B35[F4950100]    <1>       mov    esi, [csftdf_sf_mem_addr]
5605 0000AADA 8935[D08A0100]    <1>       mov    [MainProgCfg_LineOffset], esi
5606                            <1>
```

```
5607 0000AAE0 A1[CC8A0100]        <1>       mov    eax, [MainProgCfg_FileSize]
5608 0000AAE5 89C2                <1>       mov    edx, eax
5609 0000AAE7 01F2                <1>       add    edx, esi
5610                              <1>
5611                              <1> loc_mcfg_process_next_line_check:
5612 0000AAE9 89C1                <1>       mov    ecx, eax
5613                              <1>
5614 0000AAEB 803E2A              <1>       cmp    byte [esi], "*" ; Remark sign
5615 0000AAEE 7503                <1>       jne    short loc_mcfg_process_next_line
5616 0000AAF0 46                  <1>       inc    esi
5617 0000AAF1 EB17                <1>       jmp    short loc_move_mainprog_cfg_nl1
5618                              <1>
5619                              <1> loc_mcfg_process_next_line:
5620 0000AAF3 83F94F              <1>       cmp    ecx, 79
5621 0000AAF6 7605                <1>       jna    short loc_start_mainprog_cfg_process
5622                              <1>
5623 0000AAF8 B94F000000          <1>       mov    ecx, 79
5624                              <1>
5625                              <1> loc_start_mainprog_cfg_process:
5626 0000AAFD BF[8E8B0100]        <1>       mov    edi, CommandBuffer
5627                              <1>
5628                              <1> loc_move_mainprog_cfg_line:
5629 0000AB02 AC                  <1>       lodsb
5630 0000AB03 3C20                <1>       cmp    al, 20h
5631 0000AB05 720C                <1>       jb     short loc_move_mainprog_cfg_nl2
5632 0000AB07 AA                  <1>       stosb
5633 0000AB08 E2F8                <1>       loop   loc_move_mainprog_cfg_line
5634                              <1>
5635                              <1> loc_move_mainprog_cfg_nl1:
5636 0000AB0A 39D6                <1>       cmp    esi, edx ; + configuration file size
5637 0000AB0C 7312                <1>       jnb    short loc_end_of_mainprog_cfg_line
5638 0000AB0E AC                  <1>       lodsb
5639 0000AB0F 3C20                <1>       cmp    al, 20h
5640 0000AB11 73F7                <1>       jnb    short loc_move_mainprog_cfg_nl1
5641                              <1>
5642                              <1> loc_move_mainprog_cfg_nl2:
5643 0000AB13 39D6                <1>       cmp    esi, edx
5644 0000AB15 7309                <1>       jnb    short loc_end_of_mainprog_cfg_line
5645 0000AB17 8A06                <1>       mov    al, [esi]
5646 0000AB19 3C20                <1>       cmp    al, 20h
5647 0000AB1B 7703                <1>       ja     short loc_end_of_mainprog_cfg_line
5648 0000AB1D 46                  <1>       inc    esi
5649 0000AB1E EBF3                <1>       jmp    short loc_move_mainprog_cfg_nl2
5650                              <1>
5651                              <1> loc_end_of_mainprog_cfg_line:
5652 0000AB20 C60700              <1>       mov    byte [edi], 0
5653                              <1>
5654 0000AB23 8935[D08A0100]      <1>       mov    [MainProgCfg_LineOffset], esi
5655                              <1>
5656                              <1>       ; 22/11/2017
5657 0000AB29 BE[968B0100]        <1>       mov    esi, CommandBuffer + 8
5658 0000AB2E 29FE                <1>       sub    esi, edi
5659 0000AB30 7606                <1>       jna    short loc_move_mainprog_cfg_command
5660 0000AB32 30C0                <1>       xor    al, al
5661                              <1> loc_mainprog_cfg_clear_chrs:
5662 0000AB34 AA                  <1>       stosb
5663 0000AB35 4E                  <1>       dec    esi
5664 0000AB36 75FC                <1>       jnz    short loc_mainprog_cfg_clear_chrs
5665                              <1>
5666                              <1> loc_move_mainprog_cfg_command:
5667 0000AB38 BE[8E8B0100]        <1>       mov    esi, CommandBuffer
5668 0000AB3D 89F7                <1>       mov    edi, esi
5669 0000AB3F 31DB                <1>       xor    ebx, ebx
5670                              <1>       ;xor   ecx, ecx
5671 0000AB41 30C9                <1>       xor    cl, cl
5672                              <1>
5673                              <1> loc_move_mcfg_first_cmd_char:
5674 0000AB43 8A041E              <1>       mov    al, [esi+ebx]
5675 0000AB46 FEC3                <1>       inc    bl
5676 0000AB48 3C20                <1>       cmp    al, 20h
5677 0000AB4A 7712                <1>       ja     short loc_move_mcfg_cmd_capitalizing
5678 0000AB4C 7237                <1>       jb     short loc_move_mcfg_cmd_arguments_ok
5679 0000AB4E 80FB4F              <1>       cmp    bl, 79
5680 0000AB51 72F0                <1>       jb     short loc_move_mcfg_first_cmd_char
5681 0000AB53 EB30                <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
5682                              <1>
5683                              <1> loc_move_mcfg_next_cmd_char:
5684 0000AB55 8A041E              <1>       mov    al, [esi+ebx]
5685 0000AB58 FEC3                <1>       inc    bl
5686 0000AB5A 3C20                <1>       cmp    al, 20h
5687 0000AB5C 7614                <1>       jna    short loc_move_mcfg_cmd_ok
5688                              <1>
5689                              <1> loc_move_mcfg_cmd_capitalizing:
5690 0000AB5E 3C61                <1>       cmp    al, 61h ; 'a'
5691 0000AB60 7206                <1>       jb     short loc_move_mcfg_cmd_caps_ok
5692 0000AB62 3C7A                <1>       cmp    al, 7Ah ; 'z'
5693 0000AB64 7702                <1>       ja     short loc_move_mcfg_cmd_caps_ok
5694 0000AB66 24DF                <1>       and    al, 0DFh ; sub      al, 'a'-'A'
5695                              <1>
5696                              <1> loc_move_mcfg_cmd_caps_ok:
5697 0000AB68 AA                  <1>       stosb
5698 0000AB69 FEC1                <1>       inc    cl
5699 0000AB6B 80FB4F              <1>       cmp    bl, 79
5700 0000AB6E 72E5                <1>       jb     short loc_move_mcfg_next_cmd_char
5701 0000AB70 EB13                <1>       jmp    short loc_move_mcfg_cmd_arguments_ok
5702                              <1>
5703                              <1> loc_move_mcfg_cmd_ok:
5704 0000AB72 30C0                <1>       xor    al, al ; 0
5705                              <1>
5706                              <1> loc_move_mcfg_cmd_arguments:
5707 0000AB74 8807                <1>       mov    [edi], al
5708 0000AB76 47                  <1>       inc    edi
5709 0000AB77 80FB4F              <1>       cmp    bl, 79
5710 0000AB7A 7309                <1>       jnb    short loc_move_mcfg_cmd_arguments_ok
5711 0000AB7C 8A041E              <1>       mov    al, [esi+ebx]
```

```
5712 0000AB7F FEC3              <1>       inc    bl
5713 0000AB81 3C20              <1>       cmp    al, 20h
5714 0000AB83 73EF              <1>       jnb    short loc_move_mcfg_cmd_arguments
5715                            <1>
5716                            <1> loc_move_mcfg_cmd_arguments_ok:
5717 0000AB85 C60700            <1>       mov    byte [edi], 0
5718                            <1>
5719                            <1> loc_mcfg_process_cmd_interpreter:
5720 0000AB88 E825E0FFFF        <1>       call    command_interpreter
5721                            <1>
5722                            <1> loc_mcfg_ci_return_addr:
5723 0000AB8D A1[CC8A0100]      <1>       mov    eax, [MainProgCfg_FileSize]
5724 0000AB92 89C2              <1>       mov    edx, eax
5725 0000AB94 8B35[D08A0100]    <1>       mov    esi, [MainProgCfg_LineOffset]
5726 0000AB9A 01F2              <1>       add    edx, esi
5727 0000AB9C 0305[F4950100]    <1>       add    eax, [csftdf_sf_mem_addr]
5728 0000ABA2 29F0              <1>       sub    eax, esi
5729 0000ABA4 0F873FFFFFFF      <1>         ja      loc_mcfg_process_next_line_check
5730                            <1>
5731 0000ABAA E81D000000        <1>       call   mcfg_deallocate_mem
5732                            <1>
5733 0000ABAF B94F000000        <1>       mov    ecx, 79 ; 80 ?
5734 0000ABB4 BF[8E8B0100]      <1>       mov    edi, CommandBuffer
5735 0000ABB9 30C0              <1>       xor    al, al
5736 0000ABBB F3AA              <1>       rep    stosb
5737                            <1>
5738                            <1>       ; 06/05/2016
5739 0000ABBD BE[434D0100]      <1>       mov    esi, nextline
5740 0000ABC2 E89EC9FFFF        <1>       call   print_msg
5741 0000ABC7 E963D6FFFF        <1>       jmp    dos_prompt
5742                            <1>
5743                            <1> mcfg_deallocate_mem:
5744 0000ABCC A1[F4950100]      <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
5745 0000ABD1 8B0D[F8950100]    <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
5746                            <1>       ;call  deallocate_memory_block
5747                            <1>       ;retn
5748 0000ABD7 E9B6BAFFFF        <1>       jmp    deallocate_memory_block
5749                            <1>
5750                            <1> mcfg_read_file_sectors:
5751                            <1>       ; 14/04/2016
5752 0000ABDC 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
5753 0000ABE0 7669              <1>         jna  short mcfg_read_fs_file_sectors
5754                            <1>
5755                            <1> mcfg_read_fat_file_sectors:
5756                            <1>       ; return:
5757                            <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
5758                            <1>       ;   CF = 0 & EDX = 0 -> not EOF
5759                            <1>       ;   CF = 1 -> read error (error code in AL)
5760                            <1>
5761                            <1> mcfg_read_fat_file_secs_0:
5762 0000ABE2 8B15[CC8A0100]      <1>       mov    edx, [MainProgCfg_FileSize]
5763 0000ABE8 2B15[0C960100]    <1>       sub    edx, [csftdf_sf_rbytes]
5764 0000ABEE 3B15[04960100]    <1>       cmp    edx, [csftdf_r_size]
5765 0000ABF4 7306              <1>       jnb    short mcfg_read_fat_file_secs_1
5766 0000ABF6 8915[04960100]    <1>       mov    [csftdf_r_size], edx
5767                            <1>
5768                            <1> mcfg_read_fat_file_secs_1:
5769 0000ABFC A1[04960100]      <1>       mov    eax, [csftdf_r_size]
5770 0000AC01 29D2              <1>       sub    edx, edx
5771 0000AC03 0FB74E11          <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
5772 0000AC07 01C8              <1>       add    eax, ecx
5773 0000AC09 48                <1>       dec    eax
5774 0000AC0A F7F1              <1>       div    ecx
5775 0000AC0C 89C1              <1>       mov    ecx, eax ; sector count
5776 0000AC0E A1[FC950100]      <1>       mov    eax, [csftdf_sf_cluster]
5777                            <1>
5778                            <1>       ; EBX = memory block address (current)
5779                            <1>
5780 0000AC13 E88C230000        <1>       call   read_fat_file_sectors
5781 0000AC18 7230              <1>       jc     short mcfg_read_fat_file_secs_3
5782                            <1>
5783                            <1>       ; EBX = next memory address
5784                            <1>
5785 0000AC1A A1[0C960100]      <1>       mov    eax, [csftdf_sf_rbytes]
5786 0000AC1F 0305[04960100]    <1>       add    eax, [csftdf_r_size]
5787 0000AC25 8B15[CC8A0100]    <1>       mov    edx, [MainProgCfg_FileSize]
5788 0000AC2B 39D0              <1>       cmp    eax, edx
5789 0000AC2D 731B              <1>       jnb    short mcfg_read_fat_file_secs_3 ; edx > 0
5790 0000AC2F A3[0C960100]      <1>       mov    [csftdf_sf_rbytes], eax
5791                            <1>
5792 0000AC34 53                <1>       push   ebx ; *
5793                            <1>       ; get next cluster (csftdf_r_size! bytes)
5794 0000AC35 A1[FC950100]      <1>       mov    eax, [csftdf_sf_cluster]
5795 0000AC3A E837210000        <1>       call   get_next_cluster
5796 0000AC3F 5B                <1>       pop    ebx ; *
5797 0000AC40 7301              <1>       jnc    short mcfg_read_fat_file_secs_2
5798                            <1>
5799                            <1>       ;mov   eax, 17; Read error !
5800 0000AC42 C3                <1>       retn
5801                            <1>
5802                            <1> mcfg_read_fat_file_secs_2:
5803 0000AC43 29D2              <1>       sub    edx, edx ; 0
5804 0000AC45 A3[FC950100]      <1>       mov    [csftdf_sf_cluster], eax ; next cluster
5805                            <1>
5806                            <1> mcfg_read_fat_file_secs_3:
5807 0000AC4A C3                <1>       retn
5808                            <1>
5809                            <1> mcfg_read_fs_file_sectors:
5810 0000AC4B C3                <1>       retn
5811                            <1>
5812                            <1> loc_mcfg_load_fs_file:
5813 0000AC4C C3                <1>       retn
5814                            <1>
5815                            <1> load_and_execute_file:
5816                            <1>       ; 04/01/2017
```

```
                                <1>       ; 06/05/2016, 07/05/2016, 11/05/2016
                                <1>       ; 23/04/2016, 24/04/2016
                                <1>       ; 22/04/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 05/11/2011
                                <1>       ; (TRDOS v1, CMDINTR.ASM, 'cmp_cmd_run', 'cmp_cmd_external')
                                <1>       ; ('loc_run_check_filename')
                                <1>       ; 29/08/2011
                                <1>       ; 10/09/2011
                                <1>       ; INPUT->
                                <1>       ;     ESI = Path Name address (CommandBuffer address)
                                <1>       ; OUTPUT ->
                                <1>       ;     none (error message will be shown if an error will occur)
                                <1>       ;
                                <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI, EBP will be changed)
                                <1>       ;
                                <1> loc_run_check_filename:
0000AC4D 803E20                 <1>       cmp    byte [esi], 20h
0000AC50 0F822BE3FFFF           <1>       jb     loc_cmd_failed
0000AC56 7703                   <1>       ja     short loc_run_check_filename_ok
0000AC58 46                     <1>       inc    esi
0000AC59 EBF2                   <1>       jmp    short loc_run_check_filename
                                <1>
                                <1> loc_run_check_filename_ok:
0000AC5B C605[3F8B0100]00       <1>       mov    byte [CmdArgStart], 0 ; reset
0000AC62 56                     <1>       push   esi ; *
                                <1> loc_run_get_first_arg_pos:
0000AC63 46                     <1>       inc    esi
0000AC64 8A06                   <1>       mov    al, [esi]
0000AC66 3C20                   <1>       cmp    al, 20h
0000AC68 77F9                   <1>       ja     short loc_run_get_first_arg_pos
0000AC6A C60600                 <1>       mov    byte [esi], 0
                                <1> loc_run_get_external_arg_pos:
                                <1>       ; 11/05/2016
0000AC6D 46                     <1>       inc    esi
0000AC6E 8A06                   <1>       mov    al, [esi]
0000AC70 3C20                   <1>       cmp    al, 20h
0000AC72 760C                   <1>       jna    short loc_run_parse_path_name
0000AC74 89F0                   <1>       mov    eax, esi
0000AC76 2D[8E8B0100]           <1>       sub    eax, CommandBuffer
0000AC7B A2[3F8B0100]           <1>       mov    byte [CmdArgStart], al
                                <1> loc_run_parse_path_name:
0000AC80 5E                     <1>       pop    esi ; *
0000AC81 BF[7E930100]           <1>       mov    edi, FindFile_Drv
0000AC86 E8D7090000             <1>       call   parse_path_name
0000AC8B 0F82F0E2FFFF           <1>       jc     loc_cmd_failed
                                <1>
                                <1> loc_run_check_filename_exists:
0000AC91 BE[C0930100]           <1>       mov    esi, FindFile_Name
0000AC96 803E20                 <1>       cmp    byte [esi], 20h
0000AC99 0F86E2E2FFFF           <1>       jna    loc_cmd_failed
                                <1>
                                <1> loc_run_check_exe_filename_ext:
0000AC9F E890020000             <1>       call   check_prg_filename_ext
0000ACA4 0F82D7E2FFFF           <1>       jc     loc_cmd_failed
                                <1>
                                <1> loc_run_check_exe_filename_ext_ok:
0000ACAA 66A3[96960100]         <1>       mov    word [EXE_ID], ax
                                <1>
                                <1> loc_run_drv:
0000ACB0 C605[95960100]00       <1>       mov    byte [Run_Manual_Path], 0
0000ACB7 A1[D88A0100]           <1>       mov    eax, [Current_Dir_FCluster]
0000ACBC A3[90960100]           <1>         mov     [Run_CDirFC], eax
                                <1>       ;
0000ACC1 8A35[DE8A0100]         <1>       mov    dh, [Current_Drv]
0000ACC7 8835[3A920100]         <1>       mov    [RUN_CDRV], dh
                                <1>
0000ACCD 8A15[7E930100]         <1>       mov    dl, [FindFile_Drv]
0000ACD3 38F2                   <1>       cmp    dl, dh
0000ACD5 7412                   <1>       je     short loc_run_change_directory
                                <1>
0000ACD7 8005[95960100]02       <1>       add    byte [Run_Manual_Path], 2
                                <1>
0000ACDE E80BD4FFFF             <1>       call   change_current_drive
0000ACE3 0F82C3E2FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
                                <1> loc_run_change_directory:
0000ACE9 803D[7F930100]20       <1>       cmp    byte [FindFile_Directory], 20h
0000ACF0 7623                   <1>       jna    short loc_run_find_executable_file
                                <1>
0000ACF2 FE05[95960100]         <1>       inc    byte [Run_Manual_Path]
                                <1>
0000ACF8 FE05[A6400100]         <1>       inc    byte [Restore_CDIR]
                                <1>
0000ACFE BE[7F930100]           <1>       mov    esi, FindFile_Directory
0000AD03 30E4                   <1>       xor    ah, ah ; CD_COMMAND sign -> 0
0000AD05 E842030000             <1>       call   change_current_directory
0000AD0A 0F829CE2FFFF           <1>       jc     loc_run_cmd_failed
                                <1>
                                <1> loc_run_change_prompt_dir_string:
0000AD10 E857020000             <1>       call   change_prompt_dir_string
                                <1>
                                <1> loc_run_find_executable_file:
0000AD15 66C705[94960100]00-    <1>       mov    word [Run_Auto_Path], 0
0000AD1D 00                     <1>
                                <1>
                                <1> loc_run_find_executable_file_next:
0000AD1E BE[C0930100]           <1>       mov    esi, FindFile_Name
                                <1> loc_run_find_program_file_next:
0000AD23 66B80018               <1>       mov    ax, 1800h ; Except volume label and dirs
0000AD27 E865E7FFFF             <1>       call   find_first_file
                                <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
                                <1>       ; EDI = Directory Buffer Directory Entry Location
                                <1>       ; EAX = File size
0000AD2C 0F835C010000           <1>       jnc    loc_load_and_run_file
                                <1>
```

```
5921 0000AD32 3C02                <1>       cmp    al, 2 ; file not found
5922 0000AD34 0F8572E2FFFF        <1>       jne    loc_run_cmd_failed
5923                              <1>
5924 0000AD3A 66A1[96960100]      <1>       mov    ax, word [EXE_ID]
5925 0000AD40 80FC2E              <1>       cmp    ah, '.' ; File name has extension sign
5926 0000AD43 7424                <1>       je     short loc_run_check_auto_path
5927                              <1>
5928 0000AD45 08C0                <1>       or     al, al
5929 0000AD47 7520                <1>       jnz    short loc_run_check_auto_path
5930                              <1>
5931 0000AD49 80FC08              <1>       cmp    ah, 8 ; count of file name chars
5932 0000AD4C 771B                <1>       ja     short loc_run_check_auto_path
5933                              <1>
5934                              <1> loc_run_change_file_ext_to_prg:
5935 0000AD4E 0FB6DC              <1>       movzx  ebx, ah ; count of file name chars
5936 0000AD51 BE[C0930100]        <1>       mov    esi, FindFile_Name
5937 0000AD56 01F3                <1>       add    ebx, esi
5938                              <1>       ; 07/05/2016
5939 0000AD58 C7032E505247        <1>       mov    dword [ebx],  '.PRG'
5940 0000AD5E 66C705[96960100]50- <1>       mov    word [EXE_ID], 'P.'
5940 0000AD66 2E                  <1>
5941 0000AD67 EBBA                <1>       jmp    short loc_run_find_program_file_next
5942                              <1>
5943                              <1> loc_run_check_auto_path:
5944                              <1>       ; NOTE: /// 07/05/2016 ///
5945                              <1>       ; If the path is given, value of byte [Run_Manual_Path]
5946                              <1>       ; will not be ZERO. If so, file searching by using
5947                              <1>       ; Automatic Path (via 'PATH' environment variable)
5948                              <1>       ; will not be applicable, because the program file
5949                              <1>       ; is already/absolutely not found.
5950                              <1>
5951 0000AD69 A0[95960100]        <1>       mov    al, [Run_Manual_Path]
5952 0000AD6E 08C0                <1>       or     al, al
5953 0000AD70 0F850BE2FFFF        <1>       jnz    loc_cmd_failed
5954                              <1>
5955                              <1> loc_run_check_auto_path_again:
5956 0000AD76 66833D[94960100]FF  <1>       cmp    word [Run_Auto_Path], 0FFFFh
5957                              <1>              ; 0FFFFh = Not a valid run path (in ENV block)
5958 0000AD7E 0F83FDE1FFFF        <1>       jnb    loc_cmd_failed
5959                              <1>       ; xor  al, al
5960 0000AD84 BE[72410100]        <1>       mov    esi, Cmd_Path ; 'PATH'
5961 0000AD89 BF[DE8B0100]        <1>       mov    edi, TextBuffer
5962 0000AD8E E848F9FFFF          <1>       call   get_environment_string
5963 0000AD93 730E                <1>       jnc    short loc_run_chk_filename_ext_again
5964 0000AD95 66C705[94960100]FF- <1>       mov    word [Run_Auto_Path], 0FFFFh ; invalid
5964 0000AD9D FF                  <1>
5965 0000AD9E E9DEE1FFFF          <1>       jmp    loc_cmd_failed
5966                              <1>
5967                              <1> loc_run_chk_filename_ext_again:
5968 0000ADA3 89C1                <1>       mov    ecx, eax ; string length (with zero tail)
5969 0000ADA5 49                  <1>       dec    ecx ; without zero tail
5970 0000ADA6 66A1[96960100]      <1>       mov    ax, [EXE_ID]
5971 0000ADAC 80FC2E              <1>       cmp    ah, '.'
5972 0000ADAF 740E                <1>       je     short loc_run_chk_auto_path_pos
5973                              <1>
5974                              <1> loc_run_change_file_ext_to_noext_again:
5975 0000ADB1 0FB6DC              <1>       movzx  ebx, ah
5976 0000ADB4 BE[C0930100]        <1>       mov    esi, FindFile_Name
5977 0000ADB9 01F3                <1>       add    ebx, esi
5978 0000ADBB 29C0                <1>       sub    eax, eax
5979 0000ADBD 8903                <1>       mov    [ebx], eax ; 0 ; erase extension (.PRG)
5980                              <1>
5981                              <1> loc_run_chk_auto_path_pos:
5982                              <1>       ;movzx eax,  word [Run_Auto_Path]
5983 0000ADBF 66A1[94960100]      <1>       mov    ax, [Run_Auto_Path]
5984 0000ADC5 39C8                <1>       cmp    eax, ecx ; ecx = string length (except zero tail)
5985 0000ADC7 0F83B4E1FFFF        <1>       jnb    loc_cmd_failed
5986                              <1>       ;or    eax, eax
5987 0000ADCD 6609C0              <1>       or     ax, ax
5988 0000ADD0 7502                <1>       jnz    short loc_run_auto_path_pos_move
5989 0000ADD2 B005                <1>       mov    al, 5
5990                              <1>
5991                              <1> loc_run_auto_path_pos_move:
5992 0000ADD4 89FE                <1>       mov    esi, edi ; offset TextBuffer
5993 0000ADD6 01C6                <1>       add    esi, eax
5994                              <1>
5995                              <1> loc_run_auto_path_pos_space_loop:
5996 0000ADD8 AC                  <1>       lodsb
5997 0000ADD9 3C20                <1>       cmp    al, 20h
5998 0000ADDB 74FB                <1>       je     short loc_run_auto_path_pos_space_loop
5999 0000ADDD 0F829EE1FFFF        <1>       jb     loc_cmd_failed
6000 0000ADE3 AA                  <1>       stosb
6001                              <1> loc_run_auto_path_pos_move_next:
6002 0000ADE4 AC                  <1>       lodsb
6003 0000ADE5 3C3B                <1>       cmp    al, ';'
6004 0000ADE7 7414                <1>       je     short loc_run_auto_path_pos_move_last_byte
6005 0000ADE9 3C20                <1>       cmp    al, 20h
6006 0000ADEB 74F7                <1>       je     short loc_run_auto_path_pos_move_next
6007 0000ADED 7203                <1>       jb     short loc_byte_ptr_end_of_path
6008 0000ADEF AA                  <1>       stosb
6009 0000ADF0 EBF2                <1>       jmp    short loc_run_auto_path_pos_move_next
6010                              <1>
6011                              <1> loc_byte_ptr_end_of_path:
6012 0000ADF2 66C705[94960100]FF- <1>       mov    word [Run_Auto_Path], 0FFFFh ; end of path
6012 0000ADFA FF                  <1>
6013 0000ADFB EB0D                <1>       jmp    short loc_run_auto_path_move_ok
6014                              <1>
6015                              <1> loc_run_auto_path_pos_move_last_byte:
6016 0000ADFD 89F0                <1>       mov    eax, esi
6017 0000ADFF 2D[DE8B0100]        <1>       sub    eax, TextBuffer
6018 0000AE04 66A3[94960100]      <1>       mov    [Run_Auto_Path], ax ; next path position
6019                              <1>
6020                              <1> loc_run_auto_path_move_ok:
6021 0000AE0A 4F                  <1>       dec    edi
6022 0000AE0B B02F                <1>       mov    al, '/'
```

```
6023 0000AE0D 3807              <1>       cmp    [edi], al
6024 0000AE0F 7403              <1>       je     short loc_run_auto_path_move_file_name
6025 0000AE11 47                <1>       inc    edi
6026 0000AE12 8807              <1>       mov    [edi], al
6027                            <1>
6028                            <1> loc_run_auto_path_move_file_name:
6029 0000AE14 47                <1>       inc    edi
6030 0000AE15 BE[C0930100]      <1>       mov    esi, FindFile_Name
6031                            <1>
6032                            <1> loc_run_auto_path_move_fn_loop:
6033 0000AE1A AC                <1>       lodsb
6034 0000AE1B AA                <1>       stosb
6035 0000AE1C 08C0              <1>       or     al, al
6036 0000AE1E 75FA              <1>       jnz    short loc_run_auto_path_move_fn_loop
6037                            <1>
6038 0000AE20 BE[DE8B0100]      <1>       mov    esi, TextBuffer
6039 0000AE25 BF[7E930100]      <1>       mov    edi, FindFile_Drv
6040 0000AE2A E833080000        <1>       call   parse_path_name
6041 0000AE2F 0F824CE1FFFF      <1>       jc     loc_cmd_failed
6042                            <1>
6043 0000AE35 8A35[DE8A0100]    <1>       mov    dh, [Current_Drv]
6044 0000AE3B 8A15[7E930100]    <1>       mov    dl, [FindFile_Drv]
6045 0000AE41 38F2              <1>       cmp    dl, dh
6046 0000AE43 740B              <1>       je     short loc_run_change_directory_again
6047                            <1>
6048 0000AE45 E8A4D2FFFF        <1>       call   change_current_drive
6049 0000AE4A 0F825CE1FFFF      <1>       jc     loc_run_cmd_failed
6050                            <1>
6051                            <1> loc_run_change_directory_again:
6052 0000AE50 803D[7F930100]20  <1>       cmp    byte [FindFile_Directory], 20h
6053 0000AE57 761D              <1>       jna    short loc_load_executable_cdir_chk_again
6054                            <1>
6055 0000AE59 FE05[A6400100]    <1>       inc    byte [Restore_CDIR]
6056 0000AE5F BE[7F930100]      <1>       mov    esi, FindFile_Directory
6057 0000AE64 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
6058 0000AE66 E8E1010000        <1>       call   change_current_directory
6059 0000AE6B 0F823BE1FFFF      <1>       jc     loc_run_cmd_failed
6060                            <1>
6061                            <1> loc_run_chg_prompt_dir_str_again:
6062 0000AE71 E8F6000000        <1>       call   change_prompt_dir_string
6063                            <1>
6064                            <1> loc_load_executable_cdir_chk_again:
6065 0000AE76 A1[D88A0100]      <1>       mov    eax, [Current_Dir_FCluster]
6066 0000AE7B 3B05[90960100]    <1>       cmp    eax, [Run_CDirFC]
6067 0000AE81 0F8597FEFFFF      <1>       jne    loc_run_find_executable_file_next
6068 0000AE87 30C0              <1>       xor    al, al ; 0
6069 0000AE89 E9E8FEFFFF        <1>       jmp    loc_run_check_auto_path_again
6070                            <1>
6071                            <1> loc_load_and_run_file:
6072                            <1>       ; 13/11/2017
6073                            <1>       ; 04/01/2017
6074                            <1>       ; 23/04/2016
6075 0000AE8E BE[C0930100]      <1>       mov    esi, FindFile_Name
6076 0000AE93 BF[DE8B0100]      <1>       mov    edi, TextBuffer
6077                            <1>
6078                            <1>       ; 24/04/2016
6079 0000AE98 31D2              <1>       xor    edx, edx
6080 0000AE9A 668915[4A040300]  <1>       mov    word [argc], dx ; 0
6081 0000AEA1 8915[8C030300]    <1>       mov    dword [u.nread], edx ; 0
6082                            <1>
6083                            <1> loc_load_and_run_file_1:
6084 0000AEA7 AC                <1>       lodsb
6085 0000AEA8 AA                <1>       stosb
6086 0000AEA9 FF05[8C030300]    <1>       inc    dword [u.nread]
6087 0000AEAF 20C0              <1>       and    al, al
6088 0000AEB1 75F4              <1>       jnz    short loc_load_and_run_file_1
6089                            <1>
6090 0000AEB3 A0[3F8B0100]      <1>       mov    al, [CmdArgStart]
6091 0000AEB8 20C0              <1>       and    al, al
6092 0000AEBA 7445              <1>       jz     short loc_load_and_run_file_7
6093                            <1>
6094 0000AEBC 0FB6F0            <1>       movzx  esi, al ; 11/05/2016
6095 0000AEBF B950000000        <1>       mov    ecx, 80
6096 0000AEC4 29F1              <1>       sub    ecx, esi
6097 0000AEC6 81C6[8E8B0100]    <1>       add    esi, CommandBuffer
6098                            <1>
6099 0000AECC 66FF05[4A040300]  <1>       inc    word [argc] ; 11/05/2016
6100                            <1>
6101                            <1> loc_load_and_run_file_2:
6102 0000AED3 AC                <1>       lodsb
6103 0000AED4 3C20              <1>       cmp    al, 20h
6104 0000AED6 7717              <1>       ja     short loc_load_and_run_file_5
6105 0000AED8 721E              <1>       jb     short loc_load_and_run_file_6
6106                            <1>
6107                            <1> loc_load_and_run_file_3:
6108 0000AEDA 803E20            <1>       cmp    byte [esi], 20h
6109 0000AEDD 7707              <1>       ja     short loc_load_and_run_file_4
6110 0000AEDF 7217              <1>       jb     short loc_load_and_run_file_6
6111 0000AEE1 46                <1>       inc    esi
6112 0000AEE2 E2F6              <1>       loop   loc_load_and_run_file_3
6113 0000AEE4 EB12              <1>       jmp    short loc_load_and_run_file_6
6114                            <1>
6115                            <1> loc_load_and_run_file_4:
6116 0000AEE6 28C0              <1>       sub    al, al ; 0
6117 0000AEE8 66FF05[4A040300]  <1>       inc    word [argc]
6118                            <1> loc_load_and_run_file_5:
6119 0000AEEF AA                <1>       stosb
6120 0000AEF0 FF05[8C030300]    <1>       inc    dword [u.nread]
6121 0000AEF6 E2DB              <1>       loop   loc_load_and_run_file_2
6122                            <1>
6123                            <1> loc_load_and_run_file_6:
6124 0000AEF8 30C0              <1>       xor    al, al ; 0
6125 0000AEFA AA                <1>       stosb
6126 0000AEFB FF05[8C030300]    <1>       inc    dword [u.nread]
6127                            <1> loc_load_and_run_file_7:
```

```
6128 0000AF01 8807                <1>       mov    [edi], al ; 0
6129 0000AF03 66FF05[4A040300]    <1>       inc    word [argc] ; 24/04/2016
6130 0000AF0A FF05[8C030300]      <1>       inc    dword [u.nread] ; 24/04/2016
6131 0000AF10 BE[DE8B0100]        <1>       mov    esi, TextBuffer
6132 0000AF15 8B15[EC930100]      <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
6133 0000AF1B 66A1[E4930100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
6134 0000AF21 C1E010              <1>       shl    eax, 16 ; 13/11/2017
6135 0000AF24 66A1[EA930100]      <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
6136                              <1>       ; EAX = First Cluster number
6137                              <1>       ; EDX = File Size
6138                              <1>       ; ESI = Argument list address
6139                              <1>       ; [argc] = argument count
6140                              <1>       ; [u.nread] = argument list length
6141 0000AF2A E83D640000           <1>       call   load_and_run_file ; trdosk6.s
6142                              <1>         ;jc  loc_run_cmd_failed ; 04/01/2017
6143                              <1> loc_load_and_run_file_8: ; 06/05/2016
6144 0000AF2F E98BE9FFFF          <1>       jmp    loc_file_rw_restore_retn
6145                              <1>
6146                              <1> check_prg_filename_ext:
6147                              <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
6148                              <1>       ; 10/09/2011
6149                              <1>       ; (TRDOS v1, CMDINTR.ASM, 'proc_check_exe_filename_ext')
6150                              <1>       ; 14/11/2009
6151                              <1>       ; INPUT ->
6152                              <1>       ;     ESI = Dot File Name
6153                              <1>       ; OUTPUT ->
6154                              <1>       ;     cf = 0 -> EXE_ID in AL
6155                              <1>       ;     ESI = Last char + 1 position
6156                              <1>       ;     cf = 1 -> Invalid executable file name
6157                              <1>       ;     or no file name extension if AH<=8
6158                              <1>       ;     AL = Last file name char
6159                              <1>       ;     cf = 0 -> AL='P' (PRG), AL=0 (no extension)
6160                              <1>       ;
6161                              <1>       ; (Modified registers: EAX, ESI)
6162                              <1>
6163 0000AF34 30E4               <1>       xor    ah, ah
6164                              <1> loc_run_check_filename_ext:
6165 0000AF36 AC                 <1>       lodsb
6166 0000AF37 3C21               <1>       cmp    al, 21h
6167 0000AF39 7229               <1>       jb     short loc_check_exe_fn_retn
6168 0000AF3B FEC4               <1>       inc    ah
6169 0000AF3D 3C2E               <1>       cmp    al, '.'
6170 0000AF3F 75F5               <1>       jne    short loc_run_check_filename_ext
6171                              <1>
6172                              <1> loc_run_check_filename_ext_dot:
6173 0000AF41 80FC02             <1>       cmp    ah, 2 ; .??? is not valid
6174 0000AF44 88C4               <1>       mov    ah, al ; '.'
6175 0000AF46 7219               <1>       jb     short loc_check_prg_fn_retn
6176                              <1>
6177                              <1> loc_run_check_filename_ext_dot_ok:
6178 0000AF48 AC                 <1>       lodsb
6179 0000AF49 24DF               <1>       and    al, 0DFh
6180                              <1>
6181                              <1> loc_run_check_filename_ext_prg:
6182 0000AF4B 3C50               <1>       cmp    al, 'P'
6183 0000AF4D 7212               <1>       jb     short loc_check_prg_fn_retn
6184 0000AF4F 7711               <1>       ja     short loc_check_prg_fn_stc
6185 0000AF51 AC                 <1>       lodsb
6186 0000AF52 24DF               <1>       and    al, 0DFh
6187 0000AF54 3C52               <1>       cmp    al, 'R'
6188 0000AF56 750A               <1>       jne    short loc_check_prg_fn_stc
6189 0000AF58 AC                 <1>       lodsb
6190 0000AF59 24DF               <1>       and    al, 0DFh
6191 0000AF5B 3C47               <1>       cmp    al, 'G'
6192 0000AF5D 7503               <1>       jne    short loc_check_prg_fn_stc
6193                              <1>
6194 0000AF5F B050               <1>       mov    al, 'P'
6195                              <1> loc_check_prg_fn_retn:
6196 0000AF61 C3                 <1>       retn
6197                              <1>
6198                              <1> loc_check_prg_fn_stc:
6199 0000AF62 F9                 <1>       stc
6200 0000AF63 C3                 <1>       retn
6201                              <1>
6202                              <1> loc_check_exe_fn_retn:
6203 0000AF64 28C0               <1>       sub    al, al ; 0
6204 0000AF66 C3                 <1>       retn
6205                              <1>
6206                              <1> find_and_list_files:
6207 0000AF67 C3                 <1>       retn
6208                              <1> set_exec_arguments:
6209 0000AF68 C3                 <1>       retn
6210                              <1> delete_fs_directory:
6211 0000AF69 31C0               <1>       xor eax, eax
6212 0000AF6B C3                 <1>       retn
3090                                  %include 'trdosk4.s' ; 24/01/2016
   1                              <1> ; ****************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - Directory Functions : trdosk4.s
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 02/03/2021
   5                              <1> ; ----------------------------------------------------------------------------
   6                              <1> ; Beginning: 24/01/2016
   7                              <1> ; ----------------------------------------------------------------------------
   8                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                              <1> ; ----------------------------------------------------------------------------
  10                              <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
  11                              <1> ; DIR.ASM (09/10/2011)
  12                              <1> ; ****************************************************************************
  13                              <1>
  14                              <1> ; DIR.ASM  [ TRDOS KERNEL - COMMAND EXECUTER SECTION - DIRECTORY FUNCTIONS ]
  15                              <1> ; (c) 2004-2010  Erdogan TAN  [ 17/01/2004 ]  Last Update: 09/10/2011
  16                              <1> ; FILE.ASM [ FILE FUNCTIONS ] Last Update: 09/10/2011
  17                              <1>
  18                              <1> change_prompt_dir_string:
  19                              <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>
0000AF6C BE[3B920100]   <1>       mov    esi, PATH_Array
                        <1> change_prompt_dir_str: ; 05/10/2016 (call from 'set_working_path')
0000AF71 BF[E28A0100]   <1>       mov    edi, Current_Directory
0000AF76 8A25[DC8A0100] <1>       mov    ah, [Current_Dir_Level]
0000AF7C E807000000     <1>       call   set_current_directory_string
0000AF81 880D[3D8B0100] <1>       mov    [Current_Dir_StrLen], cl
                        <1>
0000AF87 C3             <1>       retn
                        <1>
                        <1> set_current_directory_string:
                        <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 27/03/2011
                        <1>       ; 09/10/2009
                        <1>       ; INPUT:
                        <1>       ;    ESI = Path Array Address
                        <1>       ;    EDI = Current Directory String Buffer
                        <1>       ;    AH = Current Directory Level
                        <1>       ; OUTPUT => EAX, EBX, ESI will be changed
                        <1>       ;    EDI will be same with input
                        <1>       ;    ECX = Current Directory String Length
                        <1>
0000AF88 57             <1>       push    edi
0000AF89 80FC00         <1>       cmp     ah, 0
0000AF8C 7652           <1>       jna    short pass_write_path
0000AF8E 83C610         <1>       add    esi, 16
0000AF91 89F3           <1>       mov    ebx, esi
                        <1> loc_write_path:
0000AF93 B908000000     <1>       mov    ecx, 8
                        <1> path_write_dirname1:
0000AF98 AC             <1>       lodsb
0000AF99 3C20           <1>       cmp    al, 20h
0000AF9B 7612           <1>       jna    short pass_write_dirname1
0000AF9D AA             <1>       stosb
0000AF9E 81FF[3C8B0100] <1>       cmp    edi, End_Of_Current_Dir_Str
0000AFA4 733A           <1>       jnb    short pass_write_path
0000AFA6 E2F0           <1>       loop   path_write_dirname1
0000AFA8 803E20         <1>       cmp    byte [esi], 20h
0000AFAB 7624           <1>       jna    short pass_write_dirname2
0000AFAD EB0A           <1>       jmp     short loc_put_dot_cont_ext
                        <1> pass_write_dirname1:
0000AFAF 89DE           <1>       mov    esi, ebx
0000AFB1 83C608         <1>       add    esi, 8
0000AFB4 803E20         <1>       cmp    byte [esi], 20h
0000AFB7 7618           <1>       jna    short pass_write_dirname2
                        <1> loc_put_dot_cont_ext:
0000AFB9 C6072E         <1>       mov    byte [edi], "."
                        <1>       ;mov   ecx, 3
0000AFBC B103           <1>       mov    cl, 3
                        <1> loc_check_dir_name_ext:
0000AFBE AC             <1>       lodsb
0000AFBF 47             <1>       inc    edi
0000AFC0 3C20           <1>       cmp    al, 20h
0000AFC2 760D           <1>       jna    short pass_write_dirname2
0000AFC4 8807           <1>       mov    [edi], al
0000AFC6 81FF[3C8B0100] <1>       cmp    edi, End_Of_Current_Dir_Str
0000AFCC 7312           <1>       jnb    short pass_write_path
0000AFCE E2EE           <1>       loop    loc_check_dir_name_ext
0000AFD0 47             <1>       inc    edi
                        <1> pass_write_dirname2:
0000AFD1 FECC           <1>       dec    ah
0000AFD3 740B           <1>       jz      short pass_write_path
0000AFD5 83C310         <1>       add    ebx, 16
0000AFD8 89DE           <1>       mov    esi, ebx
0000AFDA C6072F         <1>       mov    byte [edi],"/"
0000AFDD 47             <1>       inc    edi
0000AFDE EBB3           <1>       jmp    short loc_write_path
                        <1> pass_write_path:
0000AFE0 C60700         <1>       mov    byte [edi], 0
0000AFE3 47             <1>       inc    edi
0000AFE4 89F9           <1>       mov    ecx, edi
0000AFE6 5F             <1>       pop    edi
0000AFE7 29F9           <1>       sub    ecx, edi
                        <1>       ; ECX = Current Directory String Length
0000AFE9 C3             <1>       retn
                        <1>
                        <1> get_current_directory:
                        <1>       ; 15/10/2016
                        <1>       ; 14/02/2016
                        <1>       ; 24/01/2016 (TRDOS 386 = TRDOS v2.0)
                        <1>       ; 27/03/2011
                        <1>       ;
                        <1>       ; INPUT-> ESI = Current Directory Buffer
                        <1>       ;         DL = TRDOS Logical Dos Drive Number + 1
                        <1>       ;              (0= Default/Current Drive)
                        <1>       ;
                        <1>       ;   Note: Required dir buffer length may be <= 92 bytes
                        <1>       ;         for TRDOS (7*12 name chars + 7 slash + 0)
                        <1>       ; OUTPUT ->  ESI = Current Directory Buffer
                        <1>       ;            EAX, EBX, ECX, EDX, EDI will be changed
                        <1>       ;            CX/CL = Current Directory String Length
                        <1>       ;           DL = Drive Number (0 based)
                        <1>       ;             (If input is 0, output is current drv number)
                        <1>       ;             DH = same with input
                        <1>       ;   cf = 0 -> AL = 0
                        <1>       ;   cf = 1 -> error code in AL
                        <1>
                        <1> loc_get_current_drive_0:
0000AFEA 80FA00         <1>       cmp    dl, 0
```

```
0000AFED 7708                <1>       ja     short loc_get_current_drive_1
0000AFEF 8A15[DE8A0100]      <1>       mov    dl, [Current_Drv]
0000AFF5 EB17                <1>       jmp    short loc_get_current_drive_2
                             <1> loc_get_current_drive_1:
0000AFF7 FECA                <1>       dec    dl
0000AFF9 3A15[A5400100]      <1>       cmp    dl, [Last_DOS_DiskNo]
0000AFFF 760D                <1>       jna    short loc_get_current_drive_2
0000B001 B80F000000          <1>       mov    eax, 0Fh ; Invalid drive (Drive not ready!)
0000B006 F5                  <1>       cmc    ; stc
0000B007 C3                  <1>       retn
                             <1>
                             <1> loc_get_current_drive_not_ready_retn:
0000B008 5E                  <1>       pop    esi
                             <1>       ;mov   eax, 15
0000B009 66B80F00            <1>       mov    ax, 15 ; Drive not ready
0000B00D C3                  <1>       retn
                             <1>
                             <1> loc_get_current_drive_2:
0000B00E 31C0                <1>       xor    eax, eax
0000B010 88D4                <1>       mov    ah, dl
0000B012 56                  <1>       push   esi
0000B013 BE00010900          <1>       mov    esi, Logical_DOSDisks
0000B018 01C6                <1>       add    esi, eax
0000B01A 8A06                <1>       mov    al, [esi+LD_Name]
0000B01C 3C41                <1>       cmp    al, 'A'
0000B01E 72E8                <1>       jb     short loc_get_current_drive_not_ready_retn
                             <1>
0000B020 8A667F              <1>       mov    ah, [esi+LD_CDirLevel]
0000B023 08E4                <1>       or     ah, ah
0000B025 7506                <1>       jnz    short loc_get_current_drive_3
                             <1>
                             <1>       ;xor   ah, ah ; mov ah, 0
0000B027 8826                <1>       mov    [esi], ah
0000B029 31C9                <1>       xor    ecx, ecx
0000B02B EB1C                <1>       jmp    short loc_get_current_drive_4
                             <1>
                             <1> loc_get_current_drive_3:
0000B02D BF[3B920100]        <1>         mov     edi, PATH_Array
0000B032 57                  <1>       push   edi
0000B033 81C680000000        <1>       add    esi, LD_CurrentDirectory
0000B039 B920000000          <1>       mov    ecx, 32
0000B03E F3A5                <1>       rep    movsd
0000B040 5E                  <1>       pop    esi ; Path Array Address
0000B041 5F                  <1>       pop    edi ; pushed esi (current dir buffer offset)
                             <1>       ;
0000B042 E841FFFFFF          <1>       call   set_current_directory_string
0000B047 89FE                <1>       mov    esi, edi
                             <1>
                             <1> loc_get_current_drive_4:
0000B049 30C0                <1>       xor    al, al
0000B04B C3                  <1>       retn
                             <1>
                             <1> change_current_directory:
                             <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                             <1>       ; 19/02/2016
                             <1>       ; 11/02/2016
                             <1>       ; 10/02/2016
                             <1>       ; 08/02/2016
                             <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ; 18/09/2011 (DIR.ASM, 09/10/2011)
                             <1>       ; 04/10/2009
                             <1>       ; 2005
                             <1>       ; INPUT ->
                             <1>       ;      ESI = Directory string
                             <1>       ;      ah = CD command (CDh = save current dir string)
                             <1>       ; OUTPUT ->
                             <1>       ;      EDI = DOS Drive Description Table
                             <1>       ;      cf = 1 -> error
                             <1>       ;         EAX = Error code
                             <1>       ;      cf = 0 -> successful
                             <1>       ;         ESI = PATH_Array
                             <1>       ;         EAX = Current Directory First Cluster
                             <1>       ;
                             <1>       ; (EAX, EBX, ECX, EDX, ESI, EDI will be changed)
                             <1>
0000B04C 8825[C9920100]      <1>       mov    [CD_COMMAND], ah
0000B052 803E2F              <1>       cmp    byte [esi], '/'
0000B055 7505                <1>       jne    short loc_ccd_cdir_level
0000B057 46                  <1>       inc    esi
0000B058 30C0                <1>       xor    al, al
0000B05A EB05                <1>       jmp    short loc_ccd_parse_path_name
                             <1> loc_ccd_cdir_level:
0000B05C A0[DC8A0100]        <1>       mov    al, [Current_Dir_Level]
                             <1> loc_ccd_parse_path_name:
0000B061 88C4                <1>       mov    ah, al
0000B063 BF[3B920100]        <1>       mov    edi, PATH_Array
                             <1>
                             <1> ; Reset directory levels > cdir level
                             <1>       ; is this required !?
                             <1>       ;
                             <1>       ; Relations:
                             <1>       ; MAINPROG.ASM (pass_ccdrv_reset_cdir_FAT_fcluster)
                             <1>       ; proc_parse_dir_name,
                             <1>       ; proc_change_current_directory (this procedure)
                             <1>       ; proc_change_prompt_dir_string
                             <1>
0000B068 0FB6C8              <1>       movzx  ecx, al
0000B06B FEC1                <1>       inc    cl
0000B06D C0E104              <1>       shl    cl, 4
0000B070 01CF                <1>       add    edi, ecx
0000B072 B107                <1>       mov    cl, 7
0000B074 28C1                <1>       sub    cl, al
0000B076 C0E102              <1>       shl    cl, 2
0000B079 89C3                <1>       mov    ebx, eax
0000B07B 31C0                <1>       xor    eax, eax ; 0
```

```
0000B07D F3AB               <1>       rep    stosd
0000B07F 89D8               <1>       mov    eax, ebx
                            <1>
0000B081 BF[3B920100]       <1>       mov    edi, PATH_Array
                            <1>
0000B086 803E20             <1>       cmp    byte [esi], 20h
0000B089 F5                 <1>       cmc
0000B08A 7305               <1>       jnc    short pass_ccd_parse_dir_name
                            <1>
                            <1>              ; ESI = Path name
                            <1>              ; AL = CCD_Level
0000B08C 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:
0000B091 9C                 <1>       pushf
                            <1>
                            <1>       ;mov   [CCD_Level], al
                            <1>         ;mov [Last_Dir_Level], ah
0000B092 66A3[BF920100]     <1>       mov    [CCD_Level], ax
                            <1>
0000B098 31DB               <1>       xor    ebx, ebx
0000B09A 8A3D[DE8A0100]     <1>       mov    bh, [Current_Drv]
0000B0A0 BE00010900         <1>       mov    esi, Logical_DOSDisks
0000B0A5 01DE               <1>       add    esi, ebx
                            <1>
0000B0A7 9D                 <1>       popf
0000B0A8 720A               <1>       jc     short loc_ccd_bad_path_name_retn
                            <1>
0000B0AA 8935[BB920100]     <1>       mov    [CCD_DriveDT], esi
                            <1>
0000B0B0 3C07               <1>       cmp    al, 7
0000B0B2 7209               <1>       jb     short loc_ccd_load_child_dir
                            <1>
                            <1> loc_ccd_bad_path_name_retn:
0000B0B4 87F7               <1>       xchg   esi, edi
0000B0B6 B813000000         <1>       mov    eax, 19 ; Bad directory/path name
0000B0BB F9                 <1>       stc
                            <1> loc_ccd_retn_p:
0000B0BC C3                 <1>       retn
                            <1>
                            <1> loc_ccd_load_child_dir:
                            <1>       ; AL = CCD_Level
0000B0BD 08C0               <1>       or     al, al
0000B0BF 7467               <1>       jz     short loc_ccd_load_root_dir
                            <1>
0000B0C1 6689C1             <1>       mov    cx, ax
0000B0C4 C0E004             <1>       shl    al, 4
0000B0C7 0FB6F0             <1>       movzx  esi, al
0000B0CA 01FE               <1>       add    esi, edi  ; offset PATH_Array
                            <1>
0000B0CC 8B460C             <1>       mov    eax, [esi+12]
0000B0CF 38E9               <1>       cmp    cl, ch
0000B0D1 0F84F9000000       <1>         je      loc_ccd_load_sub_directory
0000B0D7 A3[D88A0100]       <1>       mov    [Current_Dir_FCluster], eax
                            <1>
                            <1> loc_ccd_load_child_dir_next:
0000B0DC 83C610             <1>       add    esi, 16 ; DOS DirEntry Format FileName Address
                            <1>
                            <1>       ; Directory attribute : 10h
0000B0DF B010               <1>       mov    al, 00010000b ; 10h (Attrib AND mask)
                            <1>       ;mov   ah, 11001000b ; C8h
                            <1>       ; Volume name attribute: 8h
0000B0E1 B408               <1>       mov    ah, 00001000b ; 08h (Attrib NAND, AND --> zero mask)
                            <1>
                            <1>       ;xor   cx, cx
0000B0E3 31C9               <1>       xor    ecx, ecx ; 02/03/2021
0000B0E5 E8B5010000         <1>       call   locate_current_dir_file
0000B0EA 7353               <1>       jnc    short loc_ccd_set_dir_cluster_ptr
                            <1>
                            <1>        ; 19/02/2016
                            <1>       ;mov   edi, [CCD_DriveDT]
0000B0EC 8A25[BF920100]     <1>       mov    ah, [CCD_Level]
0000B0F2 803D[C9920100]CD   <1>       cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
0000B0F9 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.)
0000B0FB 88E1               <1>       mov    cl, ah
0000B0FD 50                 <1>       push   eax
0000B0FE E8E3000000         <1>       call   loc_ccd_save_current_dir
0000B103 58                 <1>       pop    eax
                            <1> loc_ccd_load_child_dir_err:
0000B104 3C03               <1>       cmp    al, 3  ; AL = 2 => File not found error
0000B106 7202               <1>       jb     short loc_ccd_path_not_found_retn
0000B108 F9                 <1>       stc
0000B109 C3                 <1>       retn
                            <1>
                            <1> loc_ccd_path_not_found_retn:
0000B10A B003               <1>       mov    al, 3  ; Path not found
0000B10C C3                 <1>       retn
                            <1>
                            <1> loc_ccd_load_FAT_root_dir:
0000B10D 803D[DD8A0100]02   <1>       cmp    byte [Current_FATType], 2
0000B114 776B               <1>       ja     short loc_ccd_load_FAT32_root_dir
                            <1>
                            <1>       ;mov   esi, [CCD_DriveDT]
                            <1>       ;push  esi
0000B116 E8B61D0000         <1>       call   load_FAT_root_directory
                            <1>       ;pop   edi ; Dos Drv Description Table
                            <1>
0000B11B 89F7               <1>       mov    edi, esi
```

```
0000B11D BE[3B920100]        <1>       mov    esi, PATH_Array
0000B122 7298                <1>       jc     short loc_ccd_retn_p
                             <1>
0000B124 31C0                <1>       xor    eax, eax
0000B126 EB78                <1>         jmp  short loc_ccd_set_cdfc
                             <1>
                             <1> loc_ccd_load_root_dir:
0000B128 803D[DD8A0100]01    <1>       cmp    byte [Current_FATType], 1
0000B12F 73DC                <1>       jnb    short loc_ccd_load_FAT_root_dir
                             <1>
                             <1> loc_ccd_load_FS_root_dir:
0000B131 E8621E0000          <1>       call   load_FS_root_directory
0000B136 EB5C                <1>       jmp    short pass_ccd_load_FAT_sub_directory
                             <1>
                             <1> loc_ccd_load_FS_sub_directory_next:
0000B138 E85C1E0000          <1>       call   load_FS_sub_directory
0000B13D EB1F                <1>       jmp    short pass_ccd_set_dir_cluster_ptr
                             <1>
                             <1> loc_ccd_set_dir_cluster_ptr:
                             <1>       ; EDI = Directory Entry
0000B13F 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
0000B143 C1E010              <1>       shl    eax, 16
0000B146 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
                             <1>
0000B14A 8B35[BB920100]      <1>       mov    esi, [CCD_DriveDT]
0000B150 803D[DD8A0100]01    <1>       cmp    byte [Current_FATType], 1
0000B157 72DF                <1>       jb     short loc_ccd_load_FS_sub_directory_next
                             <1>       ;push  esi
0000B159 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
0000B15E BE[3B920100]        <1>       mov    esi, PATH_Array
0000B163 7264                <1>       jc     short loc_ccd_retn_c
                             <1>
0000B165 A1[09920100]        <1>       mov    eax, [DirBuff_Cluster]
                             <1>
0000B16A FE05[BF920100]      <1>       inc    byte [CCD_Level]
0000B170 0FB61D[BF920100]    <1>       movzx  ebx, byte [CCD_Level]
0000B177 C0E304              <1>       shl    bl, 4 ; * 16 (<= 128)
0000B17A 01DE                <1>       add    esi, ebx ; 19/02/2016
0000B17C 89460C              <1>       mov    [esi+12], eax
0000B17F EB1F                <1>       jmp    short loc_ccd_set_cdfc
                             <1>
                             <1> loc_ccd_load_FAT32_root_dir:
0000B181 BE[3B920100]        <1>       mov    esi, PATH_Array
0000B186 8B460C              <1>       mov    eax, [esi+12]
0000B189 8B35[BB920100]      <1>       mov    esi, [CCD_DriveDT]
                             <1>
                             <1> loc_ccd_load_FAT_sub_directory:
                             <1>       ;push  esi
0000B18F 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
0000B194 BE[3B920100]        <1>       mov    esi, PATH_Array
0000B199 722E                <1>       jc     short loc_ccd_retn_c
                             <1>
0000B19B A1[09920100]        <1>       mov    eax, [DirBuff_Cluster]
                             <1>
                             <1> loc_ccd_set_cdfc:
0000B1A0 8A0D[BF920100]      <1>       mov    cl, [CCD_Level]
0000B1A6 880D[DC8A0100]      <1>       mov    [Current_Dir_Level], cl
0000B1AC A3[D88A0100]        <1>       mov    [Current_Dir_FCluster], eax
                             <1>
0000B1B1 8A2D[C0920100]      <1>       mov    ch, [Last_Dir_Level]
0000B1B7 38E9                <1>       cmp    cl, ch
0000B1B9 0F821DFFFFFF        <1>       jb     loc_ccd_load_child_dir_next
                             <1>
0000B1BF 803D[C9920100]CD    <1>       cmp    byte [CD_COMMAND], 0CDh ;'CD' command or another
0000B1C6 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>
0000B1C8 F8                  <1>       clc
                             <1>
                             <1> loc_ccd_retn_c:
0000B1C9 8B3D[BB920100]      <1>       mov    edi, [CCD_DriveDT]
0000B1CF C3                  <1>       retn
                             <1>
                             <1> loc_ccd_load_sub_directory:
0000B1D0 8B35[BB920100]      <1>       mov    esi, [CCD_DriveDT]
0000B1D6 803D[DD8A0100]01    <1>       cmp    byte [Current_FATType], 1
0000B1DD 73B0                <1>       jnb    short loc_ccd_load_FAT_sub_directory
0000B1DF E8B51D0000          <1>       call   load_FS_sub_directory
0000B1E4 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>       ;
0000B1E6 BE[3B920100]        <1>       mov    esi, PATH_Array ; 19/02/2016
0000B1EB 8B3D[BB920100]      <1>       mov    edi, [CCD_DriveDT]
0000B1F1 57                  <1>       push   edi
0000B1F2 83C77F              <1>         add     edi, LD_CDirLevel
0000B1F5 880F                <1>       mov    [edi], cl
0000B1F7 47                  <1>       inc    edi ; LD_CurrentDirectory
0000B1F8 56                  <1>       push   esi
                             <1>       ;;mov  ecx, 32  ; always < 65536 (in this procedure)
0000B1F9 66B92000            <1>       mov    cx, 32
                             <1>       ; 02/03/2021
```

```
                                <1>       ;mov   ecx, 32
0000B1FD F3A5                   <1>       rep    movsd
                                <1>       ; Current directory has been saved to
                                <1>       ; the DOS drive description table, cdir area !
0000B1FF 5E                     <1>       pop    esi  ; PATH_Array
0000B200 5F                     <1>       pop    edi  ; Dos Drv Description Table
                                <1>
0000B201 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
0000B202 88C4                   <1>       mov    ah, al
0000B204 66A3[60930100]         <1>       mov    [PATH_CDLevel], ax
                                <1> repeat_ppdn_check_slash:
0000B20A AC                     <1>       lodsb
0000B20B 3C2F                   <1>       cmp    al, '/'
0000B20D 74FB                   <1>       je     short repeat_ppdn_check_slash
0000B20F 3C21                   <1>       cmp    al, 21h
0000B211 7219                   <1>       jb     short loc_ppdn_retn
0000B213 57                     <1>       push   edi
                                <1> loc_ppdn_get_dir_name:
0000B214 B90C000000             <1>       mov    ecx, 12
0000B219 BF[62930100]           <1>       mov    edi, Dir_File_Name
                                <1> repeat_ppdn_get_dir_name:
0000B21E AA                     <1>       stosb
0000B21F AC                     <1>       lodsb
0000B220 3C2F                   <1>       cmp    al, '/'
0000B222 740A                   <1>       je     short loc_check_level_dot_conv_dir_name
0000B224 3C20                   <1>       cmp    al, 20h
0000B226 7605                   <1>       jna    short loc_ppdn_end_of_path_scan
0000B228 E2F4                   <1>       loop   repeat_ppdn_get_dir_name
0000B22A 5F                     <1>       pop    edi
0000B22B F9                     <1>       stc
                                <1> loc_ppdn_retn:
0000B22C C3                     <1>       retn
                                <1>
                                <1> loc_ppdn_end_of_path_scan:
0000B22D 4E                     <1>       dec    esi
                                <1> loc_check_level_dot_conv_dir_name:
0000B22E 31C0                   <1>       xor    eax, eax
0000B230 AA                     <1>       stosb
0000B231 89F3                   <1>       mov    ebx, esi
0000B233 BE[62930100]           <1>       mov    esi, Dir_File_Name
0000B238 AC                     <1>       lodsb
                                <1> repeat_ppdn_name_check_dot:
0000B239 3C2E                   <1>       cmp    al, '.'
0000B23B 7509                   <1>       jne    short loc_ppdn_convert_sub_dir_name
                                <1> repeat_ppdn_name_dot_dot:
0000B23D AC                     <1>       lodsb
0000B23E 3C2E                   <1>       cmp    al, '.'
0000B240 743E                   <1>       je     short loc_ppdn_dot_dot
0000B242 3C21                   <1>       cmp    al, 21h
0000B244 7226                   <1>       jb     short pass_ppdn_convert_sub_dir_name
                                <1> loc_ppdn_convert_sub_dir_name:
0000B246 8A25[61930100]         <1>       mov    ah, [PATH_Level]
0000B24C 80FC07                 <1>       cmp    ah, 7
0000B24F 731B                   <1>       jnb    short pass_ppdn_convert_sub_dir_name
0000B251 FEC4                   <1>       inc    ah
0000B253 8825[61930100]         <1>       mov    [PATH_Level], ah
0000B259 BE[62930100]           <1>       mov    esi, Dir_File_Name
                                <1>       ;mov   edi, [PATH_Array_Ptr]
0000B25E B010                   <1>       mov    al, 16
0000B260 F6E4                   <1>       mul    ah
0000B262 8B3C24                 <1>       mov    edi, [esp]
                                <1>       ;push  edi
0000B265 01C7                   <1>       add    edi, eax
0000B267 E82B030000             <1>       call   convert_file_name
                                <1>       ;pop   edi
                                <1> pass_ppdn_convert_sub_dir_name:
0000B26C 89DE                   <1>       mov    esi, ebx
                                <1> repeat_ppdn_check_last_slash:
0000B26E AC                     <1>       lodsb
0000B26F 3C2F                   <1>       cmp    al, '/'
0000B271 74FB                   <1>       je     short repeat_ppdn_check_last_slash
0000B273 3C21                   <1>       cmp    al, 21h
0000B275 739D                   <1>       jnb    short loc_ppdn_get_dir_name
                                <1> end_of_parse_dir_name:
0000B277 5F                     <1>       pop    edi
0000B278 F5                     <1>       cmc
                                <1>       ;mov   al, [PATH_CDLevel]
                                <1>       ;mov   ah, [PATH_Level]
0000B279 66A1[60930100]         <1>       mov    ax, [PATH_CDLevel]
0000B27F C3                     <1>       retn
                                <1>
                                <1> loc_ppdn_dot_dot:
0000B280 AC                     <1>       lodsb
0000B281 3C21                   <1>       cmp    al, 21h
```

```
0000B283 73F2                <1>       jnb    short end_of_parse_dir_name
                             <1> loc_ppdn_dot_dot_prev_level:
0000B285 66A1[60930100]      <1>       mov    ax, [PATH_CDLevel]
0000B28B 80EC01              <1>       sub    ah, 1
0000B28E 80D400              <1>       adc    ah, 0
0000B291 38E0                <1>       cmp    al, ah
0000B293 7602                <1>       jna    short pass_ppdn_set_al_to_ah
0000B295 88E0                <1>       mov    al, ah
                             <1> pass_ppdn_set_al_to_ah:
0000B297 66A3[60930100]      <1>       mov    [PATH_CDLevel], ax
0000B29D EBCD                <1>       jmp    short pass_ppdn_convert_sub_dir_name
                             <1>
                             <1> locate_current_dir_file:
                             <1>       ; 20/11/2017
                             <1>       ; 14/02/2016
                             <1>       ; 13/02/2016
                             <1>       ; 10/02/2016
                             <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ; 14/08/2010
                             <1>       ; 19/09/2009
                             <1>         ; 2005
                             <1>       ; INPUT ->
                             <1>       ;      ESI = DOS DirEntry Format FileName Address
                             <1>       ;      AL = Attributes Mask
                             <1>       ;      (<AL AND EntryAttrib> must be equal to AL)
                             <1>       ;      AH = Negative Attributes Mask (If AH>0)
                             <1>       ;      (<AH AND EntryAttrib> must be ZERO)
                             <1>       ;      CH > 0 Find First Free Dir Entry or Deleted Entry
                             <1>       ;      CL = 0 -> Return the First Free Dir Entry
                             <1>       ;      CL = E5h -> Return the 1st deleted entry
                             <1>       ;      CL = FFh -> Return the 1st deleted or free entry
                             <1>       ;      CL > 0 and CL <> E5h and CL <> FFh -> Return the first
                             <1>       ;           proper entry (which fits with Atributes Masks)
                             <1>       ;      CX = 0 Find Valid File/Directory/VolumeName
                             <1>       ;      ? = Any One Char
                             <1>       ;      * = Every Chars
                             <1>       ; OUTPUT ->
                             <1>       ;      EDI = Directory Entry Address (in Directory Buffer)
                             <1>       ;      ESI = DOS DirEntry Format FileName Address
                             <1>       ;      CF = 0 -> No Error, Proper Entry,
                             <1>       ;      DL = Attributes
                             <1>       ;      DH = Previous Entry Attr (LongName Check)
                             <1>       ;      AL > 0 -> Ambiguous filename wildcard "?" used
                             <1>       ;      AH > 0 -> Ambiguous filename wildcard "*" used
                             <1>       ;      AX = 0 -> Filename full fits with directory entry
                             <1>       ;      CH = The 1st Name Char of Current Dir Entry
                             <1>       ;      CF = 1 -> Proper entry not found, Error Code in EAX/AL
                             <1>       ;      CL = 0 and CH = 0 -> Free Entry (End Of Dir)
                             <1>       ;      CL = 0 and CH = E5h -> Deleted Entry fits with filters
                             <1>       ;      CL > 0 -> Entry not found, CH invalid
                             <1>       ;      CF = 0 ->
                             <1>       ;      EBX = Current Directory Entry Index/Number (BX)
                             <1>
                             <1>       ;mov   word [DirBuff_EntryCounter], 0 ; Zero Based
                             <1>
0000B29F 8935[C3920100]      <1>       mov    [CDLF_FNAddress], esi
0000B2A5 66A3[C1920100]      <1>       mov    [CDLF_AttributesMask], ax
0000B2AB 66890D[C7920100]    <1>       mov    [CDLF_DEType], cx
                             <1>
0000B2B2 31DB                <1>       xor    ebx, ebx
0000B2B4 881D[D8920100]      <1>       mov    [PreviousAttr], bl ; 0  ; 13/02/2016
                             <1>
0000B2BA 8A3D[DE8A0100]      <1>       mov    bh, [Current_Drv]
0000B2C0 381D[04920100]      <1>       cmp    byte [DirBuff_ValidData], bl ; 0
0000B2C6 761D                <1>       jna    short loc_lcdf_reload_current_dir2
0000B2C8 8A1D[02920100]      <1>         mov     bl, [DirBuff_DRV]
0000B2CE 80EB41              <1>       sub    bl, 'A'
0000B2D1 38DF                <1>       cmp    bh, bl
0000B2D3 750E                <1>       jne    short loc_lcdf_reload_current_dir1
0000B2D5 8B15[09920100]      <1>       mov    edx, [DirBuff_Cluster]
0000B2DB 3B15[D88A0100]      <1>       cmp    edx, [Current_Dir_FCluster]
0000B2E1 7412                <1>       je     short loc_cdir_locatefile_search
                             <1>
                             <1> loc_lcdf_reload_current_dir1:
0000B2E3 30DB                <1>       xor    bl, bl
                             <1> loc_lcdf_reload_current_dir2:
0000B2E5 89DE                <1>       mov    esi, ebx
0000B2E7 81C600010900        <1>         add     esi, Logical_DOSDisks
0000B2ED E874000000          <1>       call   reload_current_directory
0000B2F2 735D                <1>       jnc    short loc_locatefile_search_again
0000B2F4 C3                  <1>       retn
                             <1>
                             <1> loc_cdir_locatefile_search:
0000B2F5 31DB                <1>       xor    ebx, ebx
0000B2F7 55                  <1>       push   ebp ; 20/11/2017
0000B2F8 E8A6000000          <1>       call   find_directory_entry
0000B2FD 5D                  <1>       pop    ebp ; 20/11/2017
0000B2FE 7349                <1>       jnc    short loc_cdir_locate_file_retn
                             <1>
                             <1> loc_locatefile_check_stc_reason:
0000B300 08ED                <1>       or     ch, ch
0000B302 7444                <1>       jz     short loc_cdir_locate_file_stc_retn
                             <1>
                             <1> loc_locatefile_check_next_entryblock:
0000B304 8A3D[DE8A0100]      <1>       mov    bh, [Current_Drv]
0000B30A 28DB                <1>       sub    bl, bl
0000B30C 0FB7F3              <1>       movzx  esi, bx
0000B30F 81C600010900        <1>         add     esi, Logical_DOSDisks
                             <1>
0000B315 803D[DC8A0100]00    <1>       cmp    byte [Current_Dir_Level], 0
0000B31C 760A                <1>       jna    short loc_locatefile_check_FAT_type
                             <1>
0000B31E 803D[DD8A0100]01    <1>       cmp    byte [Current_FATType], 1
0000B325 730A                <1>       jnb    short loc_locatefile_load_subdir_cluster
0000B327 C3                  <1>       retn
```

```
                                <1>
                                <1> loc_locatefile_check_FAT_type:
0000B328 803D[DD8A0100]03       <1>         cmp    byte [Current_FATType], 3
0000B32F 7218                   <1>         jb     short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_load_subdir_cluster:
0000B331 A1[09920100]           <1>         mov    eax, [DirBuff_Cluster]
0000B336 E83B1A0000             <1>         call   get_next_cluster
0000B33B 730D                   <1>         jnc    short loc_locatefile_next_cluster
0000B33D 09C0                   <1>         or     eax, eax
0000B33F 7507                   <1>         jnz    short loc_locatefile_drive_not_ready_read_err
0000B341 F9                     <1>         stc
                                <1> loc_locatefile_file_notfound:
0000B342 B802000000             <1>         mov    eax, 2 ; File/Directory/VolName not found
0000B347 C3                     <1>         retn
                                <1>
                                <1> loc_locatefile_drive_not_ready_read_err:
                                <1>         ;mov   eax, 17 ;Drive not ready or read error
                                <1> loc_cdir_locate_file_stc_retn:
0000B348 F5                     <1>         cmc ;stc
                                <1> loc_cdir_locate_file_retn:
0000B349 C3                     <1>         retn
                                <1>
                                <1> loc_locatefile_next_cluster:
0000B34A E80D1C0000             <1>         call   load_FAT_sub_directory
                                <1>         ;jc    short loc_locatefile_drive_not_ready_read_err
0000B34F 72F8                   <1>         jc     short loc_cdir_locate_file_retn
                                <1>
                                <1> loc_locatefile_search_again:
0000B351 8B35[C3920100]         <1>         mov    esi, [CDLF_FNAddress]
0000B357 66A1[C1920100]         <1>         mov    ax, [CDLF_AttributesMask]
0000B35D 668B0D[C7920100]       <1>         mov    cx, [CDLF_DEType]
0000B364 EB8F                   <1>         jmp    short loc_cdir_locatefile_search
                                <1>
                                <1> reload_current_directory:
                                <1>         ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 13/06/2010
                                <1>         ; 22/09/2009
                                <1>           ;
                                <1>         ; INPUT ->
                                <1>         ;      ESI = Dos drive description table address
                                <1>
                                <1>         ;mov   al, [esi+LD_FATType]
0000B366 A0[DD8A0100]           <1>         mov    al, [Current_FATType]
0000B36B 3C02                   <1>         cmp    al, 2
0000B36D 7729                   <1>         ja     short loc_reload_FAT_sub_directory
0000B36F 8A25[DC8A0100]         <1>         mov    ah, [Current_Dir_Level]
0000B375 08C0                   <1>         or     al, al
0000B377 740A                   <1>         jz     short loc_reload_FS_directory
0000B379 08E4                   <1>         or     ah, ah
0000B37B 751B                   <1>         jnz    short loc_reload_FAT_sub_directory
                                <1> loc_reload_FAT_12_16_root_directory:
0000B37D E84F1B0000             <1>         call   load_FAT_root_directory
0000B382 C3                     <1>         retn
                                <1> loc_reload_FS_directory:
0000B383 20E4                   <1>         and    ah, ah
0000B385 7506                   <1>         jnz    short loc_reload_FS_sub_directory
                                <1> loc_reload_FS_root_directory:
0000B387 E80C1C0000             <1>         call   load_FS_root_directory
0000B38C C3                     <1>         retn
                                <1> loc_reload_FS_sub_directory:
0000B38D A1[D88A0100]           <1>         mov    eax, [Current_Dir_FCluster]
0000B392 E8021C0000             <1>         call   load_FS_sub_directory
0000B397 C3                     <1>         retn
                                <1> loc_reload_FAT_sub_directory:
0000B398 A1[D88A0100]           <1>         mov    eax, [Current_Dir_FCluster]
0000B39D E8BA1B0000             <1>         call   load_FAT_sub_directory
0000B3A2 C3                     <1>         retn
                                <1>
                                <1> find_directory_entry:
                                <1>         ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                                <1>         ; 14/02/2016
                                <1>         ; 13/02/2016
                                <1>         ; 10/02/2016
                                <1>         ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 14/08/2010 (DIR.ASM, "proc_find_direntry")
                                <1>         ; 19/09/2009
                                <1>         ; 2005
                                <1>         ; INPUT ->
                                <1>         ;      ESI = Sub Dir or File Name Address
                                <1>         ;      AL = Attributes Mask
                                <1>         ;      (<AL AND EntryAttrib> must be equal to AL)
                                <1>         ;      AH = Negative Attributes Mask (If AH>0)
                                <1>         ;      (<AH AND EntryAttrib> must be ZERO)
                                <1>         ;      CH > 0 Find First Free Dir Entry or Deleted Entry
                                <1>         ;      CL = 0 -> Return the First Free Dir Entry
                                <1>         ;      CL = E5h -> Return the 1st deleted entry
                                <1>         ;      CL = FFh -> Return the 1st deleted or free entry
                                <1>         ;      CL > 0 and CL <> E5h and CL <> FFh -> Return the first
                                <1>         ;            proper entry (which fits with Atributes Masks)
                                <1>         ;      CX = 0 -> Find Valid File/Directory/VolumeName
                                <1>         ;      ? = Any One Char
                                <1>         ;      * = Every Chars
                                <1>         ;      EBX = Current Dir Entry (BX)
                                <1>         ;
                                <1>         ; OUTPUT ->
                                <1>         ;      EDI = Directory Entry Address (in DirectoryBuffer)
                                <1>         ;      ESI = Sub Dir or File Name Address
                                <1>         ;      CF = 0 -> No Error, Proper Entry,
                                <1>         ;      DL = Attributes
                                <1>         ;      DH = Previous Entry Attr (LongName Check)
                                <1>         ;      AL > 0 -> Ambiguous filename wildcard "?" used
                                <1>         ;      AH > 0 -> Ambiguous filename wildcard "*" used
                                <1>         ;      AX = 0 -> Filename full fits with directory entry
                                <1>         ;      EBX = CurrentDirEntry (BX)
```

```
                        <1>       ;      CH = The 1st Name Char of Current Dir Entry
                        <1>       ;      CF = 1 -> Proper entry not found, Error Code in AX/AL
                        <1>       ;      CL = 0 and CH = 0 -> Free Entry (End Of Dir)
                        <1>       ;      CL = 0 and CH = E5h -> Deleted Entry fits with filters
                        <1>       ;      CL > 0 -> Entry not found, CH invalid
                        <1>       ;
                        <1>       ; (EAX, EBX, ECX, EDX, EDI, EBP will be changed)
                        <1>
0000B3A3 663B1D[07920100]   <1>       cmp    bx, [DirBuff_LastEntry]
0000B3AA 0F8739010000       <1>         ja      loc_ffde_stc_retn_255
                        <1>
                        <1>       ;mov    [DirBuff_CurrentEntry], bx
                        <1>
0000B3B0 BF00000800         <1>       mov    edi, Directory_Buffer
0000B3B5 66A3[D4920100]     <1>       mov    [FDE_AttrMask], ax
                        <1>
0000B3BB 29C0               <1>       sub    eax, eax
                        <1>
                        <1>       ;;mov  [PreviousAttr], al ; 0 ;; 13/02/2016
0000B3BD 66A3[D6920100]     <1>       mov    [AmbiguousFileName], ax ; 0
                        <1>
0000B3C3 6689D8             <1>       mov    ax, bx
0000B3C6 66C1E005           <1>       shl    ax, 5 ; ; * 32 ; Directory entry size
0000B3CA 01C7               <1>       add     edi, eax
                        <1>
0000B3CC 08ED               <1>       or     ch, ch
0000B3CE 0F852D010000       <1>         jnz     loc_find_free_deleted_entry_0
                        <1>
0000B3D4 08C9               <1>       or      cl, cl
0000B3D6 0F850D010000       <1>         jnz     loc_ffde_stc_retn_255
                        <1>
                        <1> check_find_dir_entry:
0000B3DC 66A1[D4920100]     <1>       mov    ax, [FDE_AttrMask]
0000B3E2 8A2F               <1>       mov    ch, [edi]
0000B3E4 80FD00             <1>       cmp     ch, 0 ; Is it never used entry?
0000B3E7 0F8600010000       <1>       jna    loc_find_direntry_stc_retn
0000B3ED 56                 <1>       push   esi
0000B3EE 8A570B             <1>       mov    dl, [edi+0Bh] ; File attributes
0000B3F1 80FDE5             <1>       cmp    ch, 0E5h ; Is it a deleted file?
0000B3F4 746D               <1>       je     short loc_find_dir_next_entry_prevdeleted
                        <1>
0000B3F6 80FA0F             <1>       cmp     dl, 0Fh ; longname sub component check
0000B3F9 7505               <1>       jne     short loc_check_attributes_mask
0000B3FB E8EE010000         <1>       call   save_longname_sub_component
                        <1>
                        <1> loc_check_attributes_mask:
0000B400 88C6               <1>       mov    dh, al
0000B402 20D6               <1>       and    dh, dl
0000B404 38F0               <1>       cmp    al, dh
0000B406 0F85BA000000       <1>         jne     loc_find_dir_next_entry
0000B40C 20D4               <1>       and    ah, dl
0000B40E 0F85B2000000       <1>         jnz     loc_find_dir_next_entry
0000B414 80FA0F             <1>       cmp    dl, 0Fh
0000B417 751A               <1>       jne    short pass_direntry_attr_check
                        <1>
0000B419 3C0F               <1>       cmp    al, 0Fh ; AL = 0Fh -> find long name
0000B41B 0F85A5000000       <1>         jne     loc_find_dir_next_entry
                        <1>
0000B421 5E                 <1>       pop    esi
0000B422 6631C0             <1>       xor    ax, ax
0000B425 8A35[D8920100]     <1>       mov    dh, [PreviousAttr]
0000B42B 66891D[05920100]   <1>       mov    [DirBuff_CurrentEntry], bx
0000B432 C3                 <1>       retn
                        <1>
                        <1> pass_direntry_attr_check:
0000B433 89FD               <1>       mov    ebp, edi ; 14/02/2016
0000B435 B908000000         <1>       mov    ecx, 8
                        <1> loc_lodsb_find_dir:
0000B43A AC                 <1>       lodsb
0000B43B 3C2A               <1>       cmp    al, '*'
0000B43D 7508               <1>       jne    short pass_fde_ambiguous1_check
0000B43F FE05[D7920100]     <1>         inc     byte [AmbiguousFileName+1]
0000B445 EB28               <1>       jmp    short loc_check_direntry_extension
                        <1>
                        <1> pass_fde_ambiguous1_check:
0000B447 3C3F               <1>       cmp    al, '?'
0000B449 750D               <1>       jne    short pass_fde_ambiguous2_check
0000B44B FE05[D6920100]     <1>       inc    byte [AmbiguousFileName]
0000B451 803F20             <1>       cmp    byte [edi], 20h
0000B454 764E               <1>       jna    short loc_find_dir_next_entry_ebp
0000B456 EB14               <1>       jmp    short loc_scasb_find_dir_inc_di
                        <1>
                        <1> pass_fde_ambiguous2_check:
0000B458 3C20               <1>       cmp    al, 20h
0000B45A 750C               <1>       jne    short loc_scasb_find_dir
0000B45C 803F20             <1>       cmp    byte [edi], 20h
0000B45F 7543               <1>       jne    short loc_find_dir_next_entry_ebp
0000B461 EB0C               <1>       jmp    short loc_check_direntry_extension
                        <1>
                        <1> loc_find_dir_next_entry_prevdeleted:
0000B463 80CA80             <1>       or     dl, 80h  ; Bit 7 -> deleted entry sign
0000B466 EB5E               <1>       jmp    short loc_find_dir_next_entry
                        <1>
                        <1> loc_scasb_find_dir:
0000B468 3A07               <1>       cmp    al, [edi]
0000B46A 7538               <1>       jne    short loc_find_dir_next_entry_ebp
                        <1> loc_scasb_find_dir_inc_di:
0000B46C 47                 <1>       inc    edi
0000B46D E2CB               <1>       loop   loc_lodsb_find_dir
                        <1>
                        <1> loc_check_direntry_extension:
0000B46F BE08000000         <1>       mov    esi, 8
0000B474 89F7               <1>       mov    edi, esi ; 8
0000B476 033424             <1>       add    esi, [esp] ; Sub Dir or File Name Address
0000B479 01EF               <1>       add    edi, ebp
```

```
0000B47B B103               <1>       mov    cl, 3
                            <1> loc_lodsb_find_dir_ext:
0000B47D AC                 <1>       lodsb
0000B47E 3C2A               <1>       cmp    al, '*'
0000B480 7508               <1>       jne    short pass_fde_ambiguous3_check
0000B482 FE05[D7920100]     <1>       inc    byte [AmbiguousFileName+1]
0000B488 EB1E               <1>       jmp    short loc_find_dir_proper_direntry
                            <1>
                            <1> pass_fde_ambiguous3_check:
0000B48A 3C3F               <1>       cmp    al, '?'
0000B48C 750D               <1>       jne    short pass_fde_ambiguous4_check
0000B48E FE05[D6920100]     <1>       inc    byte [AmbiguousFileName]
0000B494 803F20             <1>       cmp    byte [edi], 20h
0000B497 760B               <1>       jna    short loc_find_dir_next_entry_ebp
0000B499 EB49               <1>       jmp    short loc_scasb_find_dir_ext_inc_di
                            <1>
                            <1> pass_fde_ambiguous4_check:
0000B49B 3C20               <1>       cmp    al, 20h
0000B49D 7541               <1>       jne    short loc_scasb_find_dir_ext
0000B49F 803F20             <1>       cmp    byte [edi], 20h
0000B4A2 7404               <1>       je     short loc_find_dir_proper_direntry
                            <1>
                            <1> loc_find_dir_next_entry_ebp:
0000B4A4 89EF               <1>       mov    edi, ebp ; 14/02/2016
0000B4A6 EB1E               <1>       jmp    short loc_find_dir_next_entry
                            <1>
                            <1> loc_find_dir_proper_direntry:
0000B4A8 30C9               <1>       xor    cl, cl
                            <1> loc_find_dir_proper_direntry_1:
0000B4AA 5E                 <1>       pop    esi
0000B4AB 89EF               <1>         mov     edi, ebp
0000B4AD 8A2F               <1>       mov    ch, [edi]
0000B4AF 8A570B             <1>       mov     dl, [edi+0Bh] ; Dir entry attributes
0000B4B2 66A1[D6920100]     <1>       mov    ax, [AmbiguousFileName]
                            <1> loc_find_dir_proper_direntry_2:
0000B4B8 8A35[D8920100]     <1>       mov     dh, [PreviousAttr]
0000B4BE 66891D[05920100]   <1>       mov    [DirBuff_CurrentEntry], bx
0000B4C5 C3                 <1>       retn
                            <1>
                            <1> loc_find_dir_next_entry:
0000B4C6 8815[D8920100]     <1>       mov    byte [PreviousAttr], dl ; LongName check
                            <1> loc_find_dir_next_entry_1:
0000B4CC 5E                 <1>       pop    esi
0000B4CD 83C720             <1>       add    edi, 32
                            <1>       ;inc   word [DirBuff_EntryCounter]
0000B4D0 6643               <1>       inc    bx
0000B4D2 663B1D[07920100]   <1>       cmp    bx, [DirBuff_LastEntry]
0000B4D9 770E               <1>       ja     short loc_ffde_stc_retn_255
0000B4DB E9FCFEFFFF         <1>         jmp     check_find_dir_entry
                            <1>
                            <1> loc_scasb_find_dir_ext:
0000B4E0 3A07               <1>       cmp    al, [edi]
0000B4E2 75C0               <1>       jne    short loc_find_dir_next_entry_ebp
                            <1> loc_scasb_find_dir_ext_inc_di:
0000B4E4 47                 <1>       inc    edi
0000B4E5 E296               <1>       loop    loc_lodsb_find_dir_ext
0000B4E7 EBC1               <1>       jmp    short loc_find_dir_proper_direntry_1
                            <1>
                            <1> loc_ffde_stc_retn_255:
                            <1>       ; 02/03/2021 (TRDOS 386 v2.0.3) ((BugFix))
                            <1>       ; (ECX must not be > 65535)
                            <1>       ; ((because 'loc_ccd_save_current_dir'
                            <1>       ;  sets CX to 32 for 'rep movsd'))
0000B4E9 66B9FFFF           <1>       mov    cx, 0FFFFh
                            <1>       ;xor   ecx, ecx
                            <1>       ;dec   ecx ; 0FFFFFFFFh
                            <1>       ;xor   eax, eax
                            <1> loc_find_direntry_stc_retn:
                            <1> loc_check_ffde_retn_1:
                            <1>       ;mov   ax, 2
0000B4ED B802000000         <1>       mov    eax, 2 ; File Not Found
0000B4F2 8A35[D8920100]     <1>       mov    dh, [PreviousAttr]
0000B4F8 66891D[05920100]   <1>       mov    [DirBuff_CurrentEntry], bx
0000B4FF F9                 <1>       stc
0000B500 C3                 <1>       retn
                            <1>
                            <1> loc_find_free_deleted_entry_0:
0000B501 66A1[D4920100]     <1>       mov    ax, [FDE_AttrMask]
0000B507 8A2F               <1>       mov    ch, [edi]
0000B509 8A570B             <1>       mov    dl, [edi+0Bh] ; File attributes
0000B50C 08C9               <1>       or     cl, cl
0000B50E 7407               <1>       jz     short loc_check_ffde_0_repeat
                            <1>       ;cmp   cl, 0E5h
                            <1>       ;je    short pass_loc_check_ffde_0_err
0000B510 80F9FF             <1>       cmp    cl, 0FFh
0000B513 7432               <1>       je     short loc_find_free_deleted_entry_1
0000B515 EB4D               <1>       jmp    short pass_loc_check_ffde_0_err
                            <1>
                            <1> loc_check_ffde_0_repeat:
0000B517 08ED               <1>       or     ch, ch
0000B519 7511               <1>       jnz    short loc_check_ffde_0_next
                            <1>
                            <1> loc_check_ffde_retn_2:
0000B51B 6629C0             <1>       sub    ax, ax
0000B51E 8A35[D8920100]     <1>       mov    dh, [PreviousAttr]
0000B524 66891D[05920100]   <1>       mov    [DirBuff_CurrentEntry], bx
0000B52B C3                 <1>       retn
                            <1>
                            <1> loc_check_ffde_0_next:
0000B52C 6643               <1>       inc    bx
0000B52E 83C720             <1>       add    edi, 32
                            <1>       ;inc   word [DirBuff_EntryCounter]
                            <1>
0000B531 663B1D[07920100]   <1>         cmp  bx, [DirBuff_LastEntry]
0000B538 77AF               <1>       ja     short loc_ffde_stc_retn_255
```

```
0000B53A 8815[D8920100]      <1>       mov    [PreviousAttr], dl
0000B540 8A2F                <1>       mov    ch, [edi]
0000B542 8A570B              <1>       mov    dl, [edi+0Bh] ; file attributes
0000B545 EBD0                <1>       jmp    short loc_check_ffde_0_repeat
                             <1>
                             <1> loc_find_free_deleted_entry_1:
0000B547 28D2                <1>       sub    dl, dl
                             <1> loc_find_free_deleted_entry_2:
0000B549 20ED                <1>       and    ch, ch
0000B54B 74CE                <1>       jz     short loc_check_ffde_retn_2
0000B54D 80FDE5              <1>       cmp    ch, 0E5h
0000B550 74C9                <1>       je     short loc_check_ffde_retn_2
0000B552 6643                <1>       inc    bx
0000B554 83C720              <1>       add    edi, 32
0000B557 663B1D[07920100]    <1>       cmp    bx, [DirBuff_LastEntry]
0000B55E 7789                <1>       ja     short loc_ffde_stc_retn_255
0000B560 8A2F                <1>       mov    ch, [edi]
0000B562 EBE5                <1>       jmp    short loc_find_free_deleted_entry_2
                             <1>
                             <1> pass_loc_check_ffde_0_err:
0000B564 38CD                <1>       cmp    ch, cl
0000B566 741F                <1>       je     short loc_check_ffde_attrib
                             <1>
0000B568 6643                <1>       inc    bx
0000B56A 83C720              <1>       add    edi, 32
0000B56D 663B1D[07920100]    <1>       cmp    bx, [DirBuff_LastEntry]
0000B574 0F876FFFFFFF        <1>         ja     loc_ffde_stc_retn_255
0000B57A 8815[D8920100]      <1>       mov    [PreviousAttr], dl
0000B580 8A2F                <1>       mov    ch, [edi]
0000B582 8A570B              <1>       mov    dl, [edi+0Bh]
0000B585 EBDD                <1>       jmp    short pass_loc_check_ffde_0_err
                             <1>
                             <1> loc_check_ffde_attrib:
0000B587 88C6                <1>       mov    dh, al
0000B589 20D6                <1>       and    dh, dl
0000B58B 38F0                <1>       cmp    al, dh
0000B58D 759D                <1>       jne    short loc_check_ffde_0_next
0000B58F 20D4                <1>       and    ah, dl
0000B591 7599                <1>       jnz    short loc_check_ffde_0_next
0000B593 30C9                <1>       xor    cl, cl
0000B595 EB84                <1>         jmp    loc_check_ffde_retn_2
                             <1>
                             <1> convert_file_name:
                             <1>       ; 06/03/2016
                             <1>       ; 11/02/2016
                             <1>       ; 07/02/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ; 06/10/2009
                             <1>       ; 2005
                             <1>       ;
                             <1>       ; INPUT  ->
                             <1>       ;      ESI = Dot File Name Location
                             <1>       ;      EDI = Dir Entry Format File Name Location
                             <1>       ; OUTPUT ->
                             <1>       ;      EDI = Dir Entry Format File Name Location
                             <1>       ;      ESI = Dot File Name Location (capitalized)
                             <1>       ;
                             <1>       ; (ECX, AL will be changed)
                             <1>
0000B597 56                  <1>       push   esi
0000B598 57                  <1>       push   edi
                             <1>
0000B599 B90B000000          <1>       mov    ecx, 11
0000B59E B020                <1>       mov    al, 20h
0000B5A0 F3AA                <1>       rep    stosb
                             <1>
0000B5A2 8B3C24              <1>       mov    edi, [esp]
                             <1>
0000B5A5 B10C                <1>       mov    cl, 12 ; file name length (max.)
                             <1>       ; 06/03/2016
                             <1>       ; Directory entry name limit (11 bytes) check for
                             <1>       ; 'rename_directory_entry' procedure.
                             <1>       ; (EDI points to Directory Entry)
                             <1>       ; (If the file name would not contain a dot
                             <1>       ; and file name length would be 12, this would cause to
                             <1>       ; overwrite the attributes byte of the directory entry.)
                             <1>       ;
0000B5A7 B50B                <1>       mov    ch, 11 ; directory entry's name length
                             <1> loc_check_first_dot:
0000B5A9 8A06                <1>       mov    al, [esi]
0000B5AB 3C2E                <1>       cmp    al, 2Eh
0000B5AD 750C                <1>       jne    short pass_check_first_dot
0000B5AF 8807                <1>       mov    [edi], al
0000B5B1 47                  <1>       inc    edi
0000B5B2 46                  <1>       inc    esi
0000B5B3 FEC9                <1>       dec    cl
0000B5B5 75F2                <1>       jnz    short loc_check_first_dot
                             <1>       ;;(ecx <= 12)
                             <1>       ;;loop loc_check_first_dot
0000B5B7 EB30                <1>       jmp    short stop_convert_file
                             <1>
                             <1> loc_get_fchar:
0000B5B9 8A06                <1>       mov    al, [esi]
                             <1> pass_check_first_dot:
0000B5BB 3C61                <1>       cmp    al, 61h ; 'a'
0000B5BD 7208                <1>       jb     short pass_name_capitalize
0000B5BF 3C7A                <1>       cmp    al, 7Ah ; 'z'
0000B5C1 7704                <1>       ja     short pass_name_capitalize
0000B5C3 24DF                <1>       and    al, 0DFh
0000B5C5 8806                <1>       mov    [esi], al
                             <1> pass_name_capitalize:
0000B5C7 3C21                <1>       cmp    al, 21h
0000B5C9 721E                <1>       jb     short stop_convert_file
0000B5CB 3C2E                <1>       cmp    al, 2Eh ; '.'
0000B5CD 750C                <1>       jne    short pass_dot_space
                             <1> add_dot_space:
```

```
1070 0000B5CF 80F904              <1>       cmp    cl, 4
1071 0000B5D2 760E                <1>       jna    short inc_and_loop
1072 0000B5D4 47                  <1>       inc    edi
1073 0000B5D5 FECD                <1>       dec    ch ; 06/03/2016
1074 0000B5D7 FEC9                <1>       dec    cl
1075 0000B5D9 EBF4                <1>       jmp    short add_dot_space
1076                              <1>
1077                              <1>       ;mov   al, 4
1078                              <1>       ;cmp   cl, al
1079                              <1>       ;jna   short inc_and_loop
1080                              <1>       ;sub   cl, al
1081                              <1>       ;add   edi, ecx
1082                              <1>       ;mov   cl, al
1083                              <1>       ;jmp   short inc_and_loop
1084                              <1>
1085                              <1> pass_dot_space:
1086 0000B5DB 8807                <1>       mov    [edi], al
1087                              <1> loc_after_double_dot:
1088                              <1>       ; 06/03/2016
1089 0000B5DD FECD                <1>       dec    ch ; count down for 11 bytes dir entry limit
1090 0000B5DF 740A                <1>       jz     short stop_convert_file_x
1091 0000B5E1 47                  <1>       inc    edi
1092                              <1> inc_and_loop:
1093 0000B5E2 FEC9                <1>       dec    cl ; count down for 12 bytes filename limit
1094 0000B5E4 7403                <1>       jz     short stop_convert_file
1095 0000B5E6 46                  <1>       inc    esi
1096                              <1>       ;;(ecx <= 12)
1097                              <1>       ;;loop loc_get_fchar
1098 0000B5E7 EBD0                <1>       jmp    short loc_get_fchar
1099                              <1>
1100                              <1> stop_convert_file:
1101                              <1>       ; 06/03/2016
1102 0000B5E9 30ED                <1>       xor    ch, ch
1103                              <1>       ; ECX < 256 ; 'find_first_file' -> xor cl, cl
1104                              <1> stop_convert_file_x:
1105 0000B5EB 5F                  <1>       pop    edi
1106 0000B5EC 5E                  <1>       pop    esi
1107 0000B5ED C3                  <1>       retn
1108                              <1>
1109                              <1> save_longname_sub_component:
1110                              <1>       ; 13/02/2016
1111                              <1>       ; 06/02/2016 (TRDOS 386 = TRDOS v2.0)
1112                              <1>       ; 28/02/2010
1113                              <1>       ; 17/10/2009
1114                              <1>       ; INPUT ->
1115                              <1>       ;      EDI = Directory Entry
1116                              <1>       ;      // This procedure is called
1117                              <1>       ;      // from 'find_directory_entry' procedure.
1118                              <1>       ;      // If the last entry returns with
1119                              <1>       ;      // a non-zero LongnameFound value and
1120                              <1>       ;      // if LFN_CheckSum value is equal to
1121                              <1>       ;      // the next shortname checksum,
1122                              <1>       ;      // long name is valid.
1123                              <1>       ;      // If a longname is longer than 65 bytes,
1124                              <1>       ;      // it is invalid for trdos. (>45h)
1125                              <1>
1126 0000B5EE 57                  <1>       push   edi
1127 0000B5EF 56                  <1>       push   esi
1128                              <1>       ;push  ebx
1129                              <1>       ;push  ecx
1130                              <1>       ;push  edx
1131 0000B5F0 50                  <1>       push   eax
1132                              <1>
1133 0000B5F1 29C9                <1>       sub    ecx, ecx
1134                              <1>       ;sub   eax, eax
1135 0000B5F3 B11A                <1>       mov    cl, 26
1136                              <1>
1137 0000B5F5 0FB607              <1>       movzx  eax, byte [edi] ; LDIR_Order
1138 0000B5F8 3C41                <1>       cmp    al, 41h  ; 40h (last long entry sign) + 1
1139 0000B5FA 722B                <1>       jb     short pass_pslnsc_last_long_entry
1140                              <1>
1141 0000B5FC 88C4                <1>       mov    ah, al
1142 0000B5FE 80EC40              <1>       sub    ah, 40h
1143 0000B601 8825[DA920100]      <1>       mov    [LFN_EntryLength], ah
1144                              <1>
1145 0000B607 3C45                <1>       cmp    al, 45h  ; 40h (last long entry sign) + 5
1146                              <1>              ; Max 130 byte length is usable in TRDOS
1147                              <1> ; 26*5 = 130
1148 0000B609 7753                <1>       ja     short loc_pslnsc_retn
1149                              <1>
1150 0000B60B 2407                <1>       and    al, 07h ; 0Fh
1151 0000B60D A2[D9920100]        <1>       mov    [LongNameFound], al
1152                              <1>
1153 0000B612 FEC8                <1>       dec    al
1154                              <1>       ;mov   cl, 26
1155 0000B614 F6E1                <1>       mul    cl
1156                              <1>
1157 0000B616 89C6                <1>       mov    esi, eax
1158 0000B618 01CE                <1>       add    esi, ecx
1159                              <1>              ; to make is an ASCIIZ string
1160                              <1>              ; with ax+26 bytes length
1161 0000B61A 81C6[DC920100]      <1>       add    esi, LongFileName
1162 0000B620 66C7060000          <1>       mov    word [esi], 0
1163 0000B625 EB16                <1>       jmp    short loc_pslsc_move_ldir_name2
1164                              <1>
1165                              <1> pass_pslnsc_last_long_entry:
1166 0000B627 3C04                <1>       cmp    al, 04h
1167 0000B629 7733                <1>       ja     short loc_pslnsc_retn
1168 0000B62B FE0D[D9920100]      <1>       dec    byte [LongNameFound]
1169 0000B631 3A05[D9920100]      <1>       cmp    al, [LongNameFound]
1170 0000B637 7525                <1>       jne    short loc_pslnsc_retn
1171                              <1>
1172                              <1> loc_pslsc_move_ldir_name1:
1173 0000B639 FEC8                <1>       dec    al
1174                              <1>       ;mov   cl, 26
```

```
1175 0000B63B F6E1                <1>       mul    cl
1176                              <1>
1177                              <1> loc_pslsc_move_ldir_name2:
1178 0000B63D 8A4F0D              <1>       mov    cl, [edi+0Dh] ; long name checksum
1179 0000B640 880D[DB920100]      <1>       mov    [LFN_CheckSum], cl
1180 0000B646 89FE                <1>       mov    esi, edi ; LDIR_Order
1181 0000B648 BF[DC920100]        <1>       mov    edi, LongFileName
1182 0000B64D 01C7                <1>       add    edi, eax
1183 0000B64F 46                  <1>       inc    esi
1184 0000B650 B105                <1>       mov    cl, 5 ; chars 1 to 5
1185 0000B652 F366A5              <1>       rep    movsw
1186 0000B655 83C603              <1>       add    esi, 3
1187 0000B658 A5                  <1>       movsd  ; char 6 & 7
1188 0000B659 A5                  <1>       movsd  ; char 8 & 9
1189 0000B65A A5                  <1>       movsd  ; char 10 & 11
1190 0000B65B 46                  <1>       inc    esi
1191 0000B65C 46                  <1>       inc    esi
1192 0000B65D A5                  <1>       movsd   ; char 12 & 13
1193                              <1>
1194                              <1> loc_pslnsc_retn:
1195 0000B65E 58                  <1>       pop    eax
1196                              <1>       ;pop   edx
1197                              <1>       ;pop   ecx
1198                              <1>       ;pop   ebx
1199 0000B65F 5E                  <1>       pop    esi
1200 0000B660 5F                  <1>       pop    edi
1201                              <1>
1202 0000B661 C3                  <1>       retn
1203                              <1>
1204                              <1> parse_path_name:
1205                              <1>       ; 10/02/2016
1206                              <1>       ; 08/02/2016 (TRDOS 386 = TRDOS v2.0)
1207                              <1>       ; 10/009/2011 ('proc_parse_pathname')
1208                              <1>       ; 27/11/2009
1209                              <1>       ; 05/12/2004
1210                              <1>       ;
1211                              <1>       ; INPUT ->
1212                              <1>       ;     ESI = Beginning of ASCIIZ pathname string
1213                              <1>       ;       EDI = Destination Address
1214                              <1>       ;            (which is TR-DOS FindFile data buffer)
1215                              <1>       ; OUTPUT ->
1216                              <1>       ;     CF = 1 -> Error
1217                              <1>       ;          EAX = Error Code (AL)
1218                              <1>       ;
1219                              <1>       ; (Modified registers: eax, ecx, esi, edi)
1220                              <1>
1221                              <1>       ; Clear the pathname bytes in TR-DOS Findfile data buffer
1222 0000B662 57                  <1>       push   edi
1223 0000B663 B914000000          <1>       mov    ecx, 20  ; 80 bytes
1224 0000B668 31C0                <1>       xor    eax, eax
1225 0000B66A F3AB                <1>       rep    stosd
1226 0000B66C 5F                  <1>       pop    edi
1227                              <1>
1228 0000B66D 668B06              <1>       mov    ax, [esi]
1229 0000B670 80FC3A              <1>       cmp    ah, ':'
1230 0000B673 741C                <1>       je     short loc_ppn_change_drive
1231 0000B675 A0[DE8A0100]        <1>       mov    al, [Current_Drv]
1232 0000B67A EB33                <1>       jmp    short pass_ppn_change_drive
1233                              <1>
1234                              <1> pass_ppn_cdir:
1235 0000B67C 8B35[FE930100]      <1>       mov    esi, [First_Path_Pos]
1236 0000B682 AC                  <1>       lodsb
1237                              <1> loc_ppn_get_filename:
1238 0000B683 83C741              <1>       add    edi, 65 ; FindFile_Name location
1239                              <1>       ; TRDOS Filename length must not be more than 12 bytes
1240                              <1>       ;mov   ecx, 12
1241 0000B686 B10C                <1>       mov    cl, 12
1242                              <1> loc_ppn_get_fnchar_next:
1243 0000B688 AA                  <1>       stosb
1244 0000B689 AC                  <1>       lodsb
1245 0000B68A 3C21                <1>       cmp    al, 21h
1246 0000B68C 7274                <1>       jb     short loc_ppn_clc_return
1247 0000B68E E2F8                <1>         loop    loc_ppn_get_fnchar_next
1248                              <1> loc_ppn_return:
1249 0000B690 C3                  <1>       retn
1250                              <1>
1251                              <1> loc_ppn_change_drive:
1252 0000B691 24DF                <1>       and    al, 0DFh
1253 0000B693 2C41                <1>       sub    al, 'A'; A:
1254 0000B695 726F                <1>       jc     short loc_ppn_invalid_drive
1255 0000B697 3805[A5400100]      <1>       cmp    [Last_DOS_DiskNo], al
1256 0000B69D 7267                <1>       jb     short loc_ppn_invalid_drive
1257                              <1>
1258 0000B69F 46                  <1>       inc    esi
1259 0000B6A0 46                  <1>       inc    esi
1260 0000B6A1 8A26                <1>       mov    ah, [esi]
1261 0000B6A3 80FC21              <1>       cmp    ah, 21h
1262 0000B6A6 7307                <1>       jnb    short pass_ppn_change_drive
1263                              <1>
1264                              <1> loc_ppn_cmd_failed:
1265                              <1>       ; File or directory name is not existing
1266 0000B6A8 8807                <1>       mov    [edi], al ; Drv
1267 0000B6AA 66B80100            <1>       mov    ax, 1 ; eax = 1
1268                              <1>       ; TR-DOS Error Code 01h = Bad Command Argument
1269                              <1>       ; MS-DOS Error Code 01h : Invalid Function Number
1270                              <1>       ;stc
1271                              <1>       ; (MainProg ErrMsg: "Bad command or file name!")
1272 0000B6AE C3                  <1>       retn
1273                              <1>
1274                              <1> pass_ppn_change_drive:
1275 0000B6AF 8935[FE930100]      <1>       mov    [First_Path_Pos], esi
1276 0000B6B5 C705[02940100]0000- <1>       mov    dword [Last_Slash_Pos], 0
1276 0000B6BD 0000                <1>
1277 0000B6BF AA                  <1>       stosb
1278 0000B6C0 8A06                <1>       mov    al, [esi]
```

```
                                <1> loc_scan_ppn_dslash:
0000B6C2 3C2F                   <1>         cmp     al, '/'
0000B6C4 7506                   <1>         jne     short loc_scan_next_slash_pos
0000B6C6 8935[02940100]         <1>         mov     [Last_Slash_Pos], esi
                                <1> loc_scan_next_slash_pos:
0000B6CC 46                     <1>         inc     esi
0000B6CD 8A06                   <1>         mov     al, [esi]
0000B6CF 3C20                   <1>         cmp     al, 20h
0000B6D1 77EF                   <1>         ja      short loc_scan_ppn_dslash
0000B6D3 833D[02940100]00       <1>         cmp     dword [Last_Slash_Pos], 0
0000B6DA 76A0                   <1>         jna     short pass_ppn_cdir
                                <1>
0000B6DC 8B0D[02940100]         <1>         mov     ecx, [Last_Slash_Pos]
0000B6E2 8B35[FE930100]         <1>         mov     esi, [First_Path_Pos]
0000B6E8 29F1                   <1>         sub     ecx, esi
0000B6EA 41                     <1>         inc     ecx
                                <1>         ;cmp    ecx, 64
0000B6EB 80F940                 <1>         cmp     cl, 64
0000B6EE 7715                   <1>         ja      short loc_ppn_invalid_drive_stc
                                <1>
0000B6F0 89F8                   <1>         mov     eax, edi ; Dest Dir String Location (65 byte)
0000B6F2 F3A4                   <1>         rep     movsb
                                <1>         ;mov    [edi], cl ; 0, End of Dir String
0000B6F4 8B35[02940100]         <1>         mov     esi, [Last_Slash_Pos]
0000B6FA 46                     <1>         inc     esi
0000B6FB 89C7                   <1>         mov     edi, eax
0000B6FD AC                     <1>         lodsb
0000B6FE 3C21                   <1>         cmp     al, 21h
0000B700 7381                   <1>         jnb     short loc_ppn_get_filename
                                <1> loc_ppn_clc_return:
                                <1>         ;clc
0000B702 31C0                   <1>         xor     eax, eax
0000B704 C3                     <1>         retn
                                <1>
                                <1> loc_ppn_invalid_drive_stc:
0000B705 F5                     <1>         cmc     ; stc
                                <1> loc_ppn_invalid_drive:
                                <1>         ; cf = 1
                                <1>         ; The Drive Letter/Char < "A" or > "Z"
0000B706 66B80F00               <1>         mov     ax, 0Fh
                                <1>         ; MS-DOS Error Code 0Fh = Disk Drive Invalid
                                <1>         ; (MainProg ErrMsg: "Drive not ready or read error!")
0000B70A C3                     <1>         retn
                                <1>
                                <1> find_longname:
                                <1>         ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 24/01/2010 (DIR.ASM, 'proc_find_longname')
                                <1>         ; 17/10/2009
                                <1>
                                <1>         ; INPUT ->
                                <1>         ;       ESI = DOS short file name address
                                <1>         ;       for example: "filename.ext"
                                <1>         ;
                                <1>         ; OUTPUT ->
                                <1>         ;       ESI = ASCIIZ longname address (cf = 0)
                                <1>         ;       cf = 1 -> error number returns in EAX (AL)
                                <1>         ;       AL = 0 & CF=1 -> longname not found
                                <1>         ;            the file/directory has no longname
                                <1>         ;       cf = 0 -> AL = FAT Type
                                <1>
                                <1>         ; 17/10/2009
                                <1>         ; ASCIIZ string will be returned
                                <1>         ; as LongFileName
                                <1>         ; clearing/reset is not needed
                                <1>         ;mov    ecx, 33
                                <1>         ;mov    edi, LongFileName
                                <1>         ;sub    ax, ax ; 0
                                <1>         ;rep    stosw
                                <1>
                                <1>         ;mov    byte [LongNameFound], 0
                                <1>
                                <1>         ; ESI = ASCIIZ file/directory name address
                                <1>         ;   AL = Attributes AND mask
                                <1>         ;       (Result of AND must be equal to AL)
                                <1>         ;   AH = Negative attributes mask
                                <1>         ;       (Result of AND must be ZERO)
0000B70B 66B80008               <1>         mov     ax, 0800h
                                <1>                 ; it must not be volume name or longname
0000B70F E87DDDFFFF             <1>         call    find_first_file
0000B714 7216                   <1>         jc      short loc_fln_retn
                                <1>
                                <1> loc_fln_check_FAT_Type:
0000B716 803D[DD8A0100]01       <1>         cmp     byte [Current_FATType], 1
0000B71D 7306                   <1>         jnb     short loc_fln_check_longname_yes_sign
                                <1>
0000B71F E839000000             <1>         call    get_fs_longname
0000B724 C3                     <1>         retn
                                <1>
                                <1> loc_fln_check_longname_yes_sign:
0000B725 08FF                   <1>         or      bh, bh
0000B727 7504                   <1>         jnz     short loc_fln_check_longnamefound_number
                                <1> loc_fln_longname_not_found_retn:
0000B729 31C0                   <1>         xor     eax, eax
                                <1>         ; cf = 1 & al = 0 -> longname not found
0000B72B F9                     <1>         stc
                                <1> loc_fln_retn:
0000B72C C3                     <1>         retn
                                <1>
                                <1> loc_fln_check_longnamefound_number:
                                <1>         ; 'LongNameFound' is set by
                                <1>           ; by 'save_longname_sub_component'
                                <1>         ; which is called from
                                <1>         ; 'find_directory_entry'
                                <1>         ; which is called from
                                <1>         ; 'find_first_file'
```

```
1384                                <1>       ; It must 1 if the longname is valid
1385 0000B72D 803D[D9920100]01     <1>         cmp     byte [LongNameFound], 1
1386 0000B734 75F3                 <1>       jne    short loc_fln_longname_not_found_retn
1387                                <1>
1388                                <1> loc_fln_calculate_checksum:
1389 0000B736 E813000000           <1>       call   calculate_checksum
1390                                <1>       ; AL = shortname checksum
1391                                <1>
1392                                <1> loc_fln_longname_validation:
1393                                <1>       ; 'LFN_CheckSum' has been set already
1394                                <1>       ; by 'save_longname_sub_component'
1395                                <1>       ; which is called from
1396                                <1>       ; 'find_directory_entry'
1397                                <1>       ; which is called from
1398                                <1>       ; 'find_first_file'
1399 0000B73B 3805[DB920100]      <1>       cmp    [LFN_CheckSum], al
1400 0000B741 75E6                 <1>       jne    short loc_fln_longname_not_found_retn
1401                                <1>
1402 0000B743 BE[DC920100]         <1>       mov    esi, LongFileName
1403 0000B748 A0[DD8A0100]         <1>       mov    al, [Current_FATType]
1404 0000B74D C3                   <1>       retn
1405                                <1>
1406                                <1> calculate_checksum:
1407                                <1>       ; 13/02/2016 (TRDOS 386 = TRDOS v2.0)
1408                                <1>       ; 17/10/2009 (DIR.ASM, 'proc_calculate_checksum')
1409                                <1>         ;
1410                                <1>       ; INPUT ->
1411                                <1>       ;     ESI = 11 byte DOS File Name location
1412                                <1>       ;     (in DOS Directory Entry Format)
1413                                <1>       ; OUTPUT ->
1414                                <1>       ;      AL = 8 bit checksum (CRC) value
1415                                <1>       ;
1416                                <1>       ; (Modified registers: EAX, ECX, ESI)
1417                                <1>
1418                                <1>       ; Erdogan Tan [ 17-10-2009 ]
1419                                <1>       ;  'ror al, 1' instruction
1420                                <1>
1421                                <1>       ; Erdogan Tan [ 20-06-2004 ]
1422                                <1>       ; This 8086 assembly code is an original code
1423                                <1>       ; which is adapted from C code in
1424                                <1>       ; Microsoft FAT32 File System Specification
1425                                <1>       ; Version 1.03, December 6, 2000
1426                                <1>       ; Page 28
1427                                <1>
1428 0000B74E 30C0                 <1>       xor    al, al
1429 0000B750 B90B000000           <1>       mov    ecx, 11
1430                                <1> loc_next_sum:
1431                                <1>       ;xor   ah, ah
1432                                <1>       ;test  al, 1
1433                                <1>       ;jz    short pass_ah_80h
1434                                <1>       ;mov   ah, 80h
1435                                <1> ;pass_ah_80h:
1436                                <1>       ;shr   al, 1
1437 0000B755 D0C8                 <1>       ror    al, 1 ; 17/10/2009
1438 0000B757 0206                 <1>       add    al, [esi]
1439 0000B759 46                   <1>       inc    esi
1440                                <1>       ;add   al, ah
1441 0000B75A E2F9                 <1>       loop   loc_next_sum
1442 0000B75C C3                   <1>       retn
1443                                <1>
1444                                <1> get_fs_longname:
1445                                <1>       ; temporary (13/02/2016)
1446 0000B75D 31C0                 <1>       xor eax, eax
1447 0000B75F F9                   <1>       stc
1448 0000B760 C3                   <1>       retn
1449                                <1>
1450                                <1> make_sub_directory:
1451                                <1>       ; 16/10/2016
1452                                <1>       ; 02/03/2016, 03/03/2016
1453                                <1>       ; 26/02/2016, 27/02/2016
1454                                <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
1455                                <1>       ; 01/08/2011 (DIR.ASM, 'proc_make_directory')
1456                                <1>       ; 10/07/2010
1457                                <1>       ; INPUT ->
1458                                <1>       ;     ESI = ASCIIZ Directory Name
1459                                <1>       ;     CL = Directory Attributes
1460                                <1>       ; OUTPUT ->
1461                                <1>       ;     EAX = New sub dir's first cluster
1462                                <1>       ;     ESI = Logical Dos Drv Descr. Table Addr.
1463                                <1>       ;     CF = 1 -> error code in AL (EAX)
1464                                <1>
1465                                <1>       ;test  cl, 10h  ; directory
1466                                <1>       ;jz    short loc_make_directory_access_denied
1467                                <1>       ;test  cl, 08h ; volume name
1468                                <1>       ;jnz   short loc_make_directory_access_denied
1469                                <1>
1470 0000B761 80E107               <1>       and    cl, 07h
1471 0000B764 880D[58940100]       <1>       mov    byte [mkdir_attrib], cl
1472                                <1>
1473 0000B76A 56                   <1>       push   esi
1474 0000B76B 31DB                 <1>       xor    ebx, ebx
1475 0000B76D 8A3D[DE8A0100]       <1>       mov    bh, [Current_Drv]
1476 0000B773 BE00010900           <1>       mov    esi, Logical_DOSDisks
1477 0000B778 01DE                 <1>       add    esi, ebx
1478 0000B77A 5B                   <1>       pop    ebx
1479                                <1>
1480                                <1>       ; 10/07/2010 -> 1st writable disk check for trdos
1481                                <1>       ; LD_DiskType = 0 for write protection (read only)
1482 0000B77B 807E0101             <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
1483 0000B77F 730B                 <1>       jnb    short loc_mkdir_check_file_sytem
1484                                <1>       ; 16/10/2016 (13h -> 30)
1485 0000B781 B81E000000           <1>       mov    eax, 30 ; 'Disk write-protected' error
1486 0000B786 BA00000000           <1>       mov    edx, 0
1487                                <1>       ; err retn: EDX = 0, EBX = Dir name offset
1488                                <1>       ;ESI = Logical DOS drive description table address
```

```
1489 0000B78B C3                <1>       retn
1490                            <1>
1491                            <1> ;loc_make_directory_access_denied:
1492                            <1>       ;mov   ax, 05h ; access denied (invalid attributes input)
1493                            <1>       ;stc
1494                            <1>       ;retn
1495                            <1>
1496                            <1> loc_mkdir_check_file_sytem:
1497 0000B78C 807E0301          <1>       cmp    byte [esi+LD_FATType], 1
1498 0000B790 730B              <1>       jnb    short loc_mkdir_check_free_sectors
1499                            <1>
1500                            <1> loc_make_fs_directory:
1501 0000B792 A1[D88A0100]      <1>       mov    eax, [Current_Dir_FCluster]
1502                            <1>       ; EAX = Parent directory DDT Address
1503                            <1>       ; ESI = Logical DOS Drive DT Address
1504                            <1>       ; EBX = Directory name offset (as ASCIIZ name)
1505 0000B797 E8D5150000        <1>       call   make_fs_directory
1506 0000B79C C3                <1>       retn
1507                            <1>
1508                            <1> loc_mkdir_check_free_sectors:
1509 0000B79D 0FB64613          <1>         movzx   eax, byte [esi+LD_BPB+SecPerClust]
1510 0000B7A1 8B4E74            <1>       mov    ecx, [esi+LD_FreeSectors]
1511 0000B7A4 39C1              <1>       cmp    ecx, eax
1512 0000B7A6 7255              <1>       jb     short loc_mkdir_insufficient_disk_space
1513                            <1>
1514                            <1> loc_make_fat_directory:
1515 0000B7A8 891D[48940100]    <1>       mov    [mkdir_DirName_Offset], ebx
1516 0000B7AE 890D[54940100]    <1>       mov    [mkdir_FreeSectors], ecx
1517                            <1>
1518                            <1>       ;mov   al, [esi+LD_BPB+SecPerClust]
1519 0000B7B4 A2[5A940100]      <1>       mov    byte [mkdir_SecPerClust], al
1520                            <1>
1521                            <1> loc_mkdir_gffc_1:
1522 0000B7B9 E80F180000        <1>       call   get_first_free_cluster
1523 0000B7BE 722A              <1>       jc     short loc_mkdir_gffc_retn
1524                            <1>
1525                            <1> ;loc_mkdir_gffc_1_cont:
1526                            <1>       ;cmp   eax, 2
1527                            <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
1528                            <1>
1529                            <1> ;loc_mkdir_gffc_1_save_fcluster:
1530 0000B7C0 A3[4C940100]      <1>       mov    [mkdir_FFCluster], eax
1531                            <1>
1532                            <1> loc_mkdir_locate_ffe:
1533                            <1>       ; Current directory fcluster <> Directory buffer cluster
1534                            <1>       ; Current directory will be reloaded by
1535                            <1>       ; 'locate_current_dir_file' procedure
1536                            <1>       ;
1537                            <1>       ; ESI = Logical DOS Drive Description Table Address
1538                            <1>       ;push  esi ; 27/02/2016
1539 0000B7C5 31C0              <1>       xor    eax, eax
1540 0000B7C7 89C1              <1>         mov  ecx, eax
1541 0000B7C9 6649              <1>       dec    cx ; FFFFh
1542                            <1>       ; CX = FFFFh -> find first deleted or free entry
1543                            <1>       ; ESI would be ASCIIZ filename address if the call
1544                            <1>       ; would not be for first free or deleted dir entry
1545 0000B7CB E8CFFAFFFF        <1>       call   locate_current_dir_file
1546 0000B7D0 734C              <1>       jnc    short loc_mkdir_set_ff_dir_entry_1
1547                            <1>       ;pop   esi
1548                            <1>       ; ESI = Logical DOS Drive Description Table Address
1549 0000B7D2 83F802            <1>       cmp    eax, 2  ; cmp al, 2 ; File/Dir not found !
1550 0000B7D5 752B              <1>       jne    short loc_mkdir_stc_return
1551                            <1>
1552                            <1> loc_mkdir_add_new_cluster:
1553 0000B7D7 3805[DD8A0100]    <1>       cmp    byte [Current_FATType], al ; 2
1554                            <1>       ;cmp   byte ptr [esi+LD_FATType], 2
1555 0000B7DD 770C              <1>       ja     short loc_mkdir_add_new_cluster_check_fsc
1556 0000B7DF 803D[DC8A0100]01  <1>       cmp    byte [Current_Dir_Level], 1
1557                            <1>       ;cmp   byte [esi+LD_CDirLevel], 1
1558 0000B7E6 7303              <1>       jnb    short loc_mkdir_add_new_cluster_check_fsc
1559                            <1>
1560 0000B7E8 B00C              <1>       mov    al, 12 ; No more files
1561                            <1> loc_mkdir_gffc_retn:
1562 0000B7EA C3                <1>       retn
1563                            <1>
1564                            <1> loc_mkdir_add_new_cluster_check_fsc:
1565 0000B7EB 8B0D[54940100]    <1>       mov    ecx, [mkdir_FreeSectors]
1566                            <1>       ;movzx eax, byte [mkdir_SecPerClust]
1567 0000B7F1 A0[5A940100]      <1>       mov    al, [mkdir_SecPerClust]
1568 0000B7F6 66D1E0            <1>       shl    ax, 1 ; AX = 2 * AX
1569 0000B7F9 39C1              <1>       cmp    ecx, eax
1570 0000B7FB 7350              <1>       jnb    short loc_mkdir_add_new_subdir_cluster
1571                            <1>
1572                            <1> loc_mkdir_insufficient_disk_space:
1573                            <1>       ;mov   edx, ecx
1574                            <1> ;loc_mkdir_gffc_insufficient_disk_space:
1575 0000B7FD 66B82700          <1>       mov    ax, 27h ; MSDOS err => insufficient disk space
1576                            <1>       ; err retn: EDX = Free sectors, EBX = Dir name offset
1577                            <1>         ; ESI -> Dos drive description table address
1578                            <1>       ;; ecx = edx
1579                            <1>       ;
1580 0000B801 C3                <1>       retn
1581                            <1>
1582                            <1> loc_mkdir_stc_return:
1583 0000B802 F9                <1>       stc
1584 0000B803 C3                <1>       retn
1585                            <1>
1586                            <1> loc_mkdir_gffc_2:
1587 0000B804 E8C4170000        <1>       call   get_first_free_cluster
1588 0000B809 72DF              <1>       jc     short loc_mkdir_gffc_retn
1589                            <1>
1590                            <1> ;loc_mkdir_gffc_1_cont:
1591                            <1>       ;cmp   eax, 2
1592                            <1>       ;jb    short loc_mkdir_gffc_insufficient_disk_space
1593                            <1>
```

```
1594                                <1> ;loc_mkdir_gffc_2_save_fcluster:
1595 0000B80B A3[4C940100]         <1>         mov    [mkdir_FFCluster], eax
1596                                <1>
1597 0000B810 A1[50940100]         <1>         mov    eax, [mkdir_LastDirCluster]
1598                                <1>
1599 0000B815 E842170000           <1>         call   load_FAT_sub_directory
1600 0000B81A 72CE                 <1>         jc     short loc_mkdir_gffc_retn
1601                                <1>
1602 0000B81C 31FF                 <1>         xor    edi, edi
1603                                <1> loc_mkdir_set_ff_dir_entry_1:
1604                                <1>         ; 27/02/2016
1605 0000B81E 56                   <1>         push   esi ; Logical DOS Drv Desc. Tbl. address
1606                                <1>         ; EDI = Directory Entry Address
1607 0000B81F 8B35[48940100]       <1>         mov    esi, [mkdir_DirName_Offset]
1608 0000B825 A1[4C940100]         <1>         mov    eax, [mkdir_FFCluster]
1609                                <1>
1610 0000B82A 66B91000             <1>         mov    cx, 10h      ; CL = Directory attribute
1611                                <1>                             ; CH = 0 -> File size is 0
1612 0000B82E 0A0D[58940100]       <1>         or     cl, [mkdir_attrib] ; S, H, R
1613 0000B834 E8B0010000           <1>         call   make_directory_entry
1614                                <1>
1615 0000B839 5E                   <1>         pop    esi
1616                                <1>
1617 0000B83A C605[04920100]02     <1>         mov    byte [DirBuff_ValidData], 2
1618 0000B841 E880020000           <1>         call   save_directory_buffer
1619 0000B846 0F83DA000000         <1>           jnc     loc_mkdir_set_ff_dir_entry_2
1620                                <1>
1621                                <1> loc_mkdir_return:
1622 0000B84C C3                   <1>         retn
1623                                <1>
1624                                <1> loc_mkdir_add_new_subdir_cluster:
1625 0000B84D 8B15[09920100]       <1>         mov    edx, [DirBuff_Cluster]
1626 0000B853 8915[50940100]       <1>         mov    [mkdir_LastDirCluster], edx
1627                                <1>
1628 0000B859 A1[4C940100]         <1>         mov    eax, [mkdir_FFCluster]
1629 0000B85E E8F9160000           <1>         call   load_FAT_sub_directory
1630 0000B863 72E7                 <1>         jc     short loc_mkdir_return
1631                                <1>         ; eax = 0
1632                                <1>         ; ecx =  directory buffer sector count (<= 128)
1633                                <1>
1634                                <1> pass_mkdir_add_new_subdir_cluster:
1635 0000B865 29FF                 <1>         sub    edi, edi ; 0
1636                                <1>         ;mov   al, 128 ; double word
1637                                <1>         ;mul   ecx ; ecx =  directory buffer sector count
1638                                <1>         ;mov   ecx, eax
1639                                <1>         ;shl   cx, 7 ; 128 * sector count
1640 0000B867 668B4611              <1>         mov    ax, [esi+LD_BPB+BytesPerSec] ; 512
1641 0000B86B 66C1E802             <1>         shr    ax, 2 ; 'byte count / 4' for 'stosd'
1642 0000B86F 66F7E1               <1>         mul    cx ; max = 128*(512/4) -> 16384 (stosd)
1643 0000B872 6689C1               <1>         mov    cx, ax
1644 0000B875 6629C0               <1>         sub    ax, ax ; 0
1645 0000B878 F3AB                 <1>         rep    stosd ; clear directory buffer
1646                                <1>
1647 0000B87A C605[04920100]02     <1>         mov    byte [DirBuff_ValidData], 2
1648 0000B881 E840020000           <1>         call   save_directory_buffer
1649 0000B886 72C4                 <1>         jc     short loc_mkdir_return
1650                                <1>
1651                                <1> loc_mkdir_save_added_cluster:
1652 0000B888 A1[50940100]         <1>         mov    eax, [mkdir_LastDirCluster]
1653 0000B88D 8B0D[4C940100]       <1>         mov    ecx, [mkdir_FFCluster]
1654                                <1>         ; 01/03/2016
1655 0000B893 31D2                 <1>         xor    edx, edx
1656 0000B895 8915[FA910100]       <1>         mov    [FAT_ClusterCounter], edx ; 0 ; reset
1657 0000B89B E800180000           <1>         call   update_cluster
1658 0000B8A0 7304                 <1>         jnc    short loc_mkdir_save_fat_buffer_0
1659 0000B8A2 09C0                 <1>         or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1660 0000B8A4 7518                 <1>         jnz    short loc_mkdir_save_fat_buffer_stc_retn
1661                                <1>
1662                                <1> loc_mkdir_save_fat_buffer_0:
1663 0000B8A6 A1[4C940100]         <1>         mov    eax, [mkdir_FFCluster]
1664 0000B8AB A3[50940100]         <1>         mov    [mkdir_LastDirCluster], eax
1665                                <1>
1666 0000B8B0 31C9                 <1>         xor    ecx, ecx
1667 0000B8B2 49                   <1>         dec    ecx ; FFFFFFFFh
1668                                <1>         ; ESI = Logical DOS Drive Description Table address
1669 0000B8B3 E8E8170000           <1>         call   update_cluster
1670 0000B8B8 731A                 <1>         jnc    short loc_mkdir_save_fat_buffer_1
1671 0000B8BA 09C0                 <1>         or     eax, eax
1672 0000B8BC 7416                 <1>         jz     short loc_mkdir_save_fat_buffer_1
1673                                <1>
1674                                <1> loc_mkdir_save_fat_buffer_stc_retn:
1675                                <1>         ; 01/03/2016
1676 0000B8BE 803D[FA910100]01     <1>         cmp    byte [FAT_ClusterCounter], 1
1677 0000B8C5 720C                 <1>         jb     short loc_mkdir_save_fat_buffer_retn
1678                                <1>
1679 0000B8C7 66BB00FF             <1>         mov    bx, 0FF00h ; recalculate free space (BL = 0)
1680                                <1>                           ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
1681 0000B8CB 50                   <1>         push   eax
1682 0000B8CC E8211B0000           <1>         call   calculate_fat_freespace
1683 0000B8D1 58                   <1>         pop    eax
1684 0000B8D2 F9                   <1>         stc
1685                                <1> loc_mkdir_save_fat_buffer_retn:
1686 0000B8D3 C3                   <1>         retn
1687                                <1>
1688                                <1> loc_mkdir_save_fat_buffer_1:
1689                                <1>         ; byte [FAT_BuffValidData] = 2
1690 0000B8D4 E8841A0000           <1>         call   save_fat_buffer
1691 0000B8D9 72E3                 <1>         jc     short loc_mkdir_save_fat_buffer_stc_retn
1692                                <1>
1693                                <1>         ; 01/03/2016
1694 0000B8DB 803D[FA910100]01     <1>         cmp    byte [FAT_ClusterCounter], 1
1695 0000B8E2 721B                 <1>         jb     short loc_mkdir_save_fat_buffer_2
1696                                <1>
1697                                <1>         ; ESI = Logical DOS Drive Description Table address
1698 0000B8E4 A1[FA910100]         <1>         mov    eax, [FAT_ClusterCounter]
```

```
1699 0000B8E9 66BB01FF            <1>       mov   bx, 0FF01h ; add free clusters
1700 0000B8ED E8001B0000          <1>       call  calculate_fat_freespace
1701                              <1>
1702                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
1703                              <1>       ;jnz   short loc_mkdir_save_fat_buffer_2
1704                              <1>
1705                              <1>       ; ecx > 0 -> Recalculation is needed
1706 0000B8F2 09C9                <1>       or    ecx, ecx
1707 0000B8F4 7409                <1>       jz    short loc_mkdir_save_fat_buffer_2
1708                              <1>
1709 0000B8F6 66BB00FF            <1>       mov   bx, 0FF00h ; ; recalculate free space
1710 0000B8FA E8F31A0000          <1>       call  calculate_fat_freespace
1711                              <1>
1712                              <1> loc_mkdir_save_fat_buffer_2:
1713 0000B8FF C605[5B940100]01    <1>       mov   byte [mkdir_add_new_cluster], 1
1714 0000B906 E9C4000000          <1>       jmp   loc_mkdir_upd_parent_dir_lmdt
1715                              <1>
1716                              <1> loc_mkdir_update_sub_dir_cluster:
1717 0000B90B A1[4C940100]        <1>       mov   eax, [mkdir_FFCluster]
1718 0000B910 29C9                <1>       sub   ecx, ecx ; 0
1719                              <1>       ; 01/03/2016
1720 0000B912 890D[FA910100]      <1>       mov   [FAT_ClusterCounter], ecx ; 0 ; Reset
1721 0000B918 49                  <1>       dec   ecx ; 0FFFFFFFFh
1722                              <1>
1723                              <1>       ; ESI = Logical DOS Drive Desciption Table address
1724 0000B919 E882170000          <1>       call  update_cluster
1725 0000B91E 7379                <1>       jnc   short loc_mkdir_save_fat_buffer_3
1726 0000B920 09C0                <1>       or    eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1727 0000B922 7475                <1>       jz    short loc_mkdir_save_fat_buffer_3
1728                              <1>       ; 01/03/2016
1729 0000B924 EB98                <1>       jmp   short loc_mkdir_save_fat_buffer_stc_retn
1730                              <1>
1731                              <1> loc_mkdir_set_ff_dir_entry_2:
1732                              <1>       ; ESI = Logical DOS Drive Description Table address
1733 0000B926 A1[4C940100]        <1>       mov   eax, [mkdir_FFCluster]
1734                              <1>       ; Load disk sectors as a directory cluster
1735 0000B92B E82C160000          <1>       call  load_FAT_sub_directory
1736 0000B930 7266                <1>       jc    short retn_make_fat_directory
1737                              <1>
1738                              <1>       ; eax = 0
1739                              <1>       ; ecx =  directory buffer sector count (<= 128)
1740                              <1>
1741 0000B932 BF40000800          <1>       mov   edi, Directory_Buffer + 64 ; 26/02/2016
1742                              <1>
1743                              <1>       ; 02/03/2016
1744 0000B937 668B4611            <1>       mov   ax, [esi+LD_BPB+BytesPerSec] ; 512
1745 0000B93B 66C1E802            <1>       shr   ax, 2 ; 'byte count / 4' for 'stosd'
1746 0000B93F F7E1                <1>       mul   ecx
1747 0000B941 89C1                <1>       mov   ecx, eax
1748 0000B943 6629C0              <1>       sub   ax, ax
1749 0000B946 F3AB                <1>       rep   stosd
1750                              <1>
1751                              <1>       ;;mov  al, 128 ; double word
1752                              <1>       ;;mul  ecx ; ecx =  directory buffer sector count
1753                              <1>       ;;mov  ecx, eax
1754                              <1>       ;shl   cx, 7 ; 128 * sector count
1755                              <1>       ;;sub  eax, eax
1756                              <1>       ;;sub  al, al ; 0
1757                              <1>       ;rep   stosd ; clear directory buffer
1758                              <1>
1759 0000B948 BF00000800          <1>       mov   edi, Directory_Buffer ; 26/02/2016
1760                              <1>
1761 0000B94D 56                  <1>       push  esi
1762                              <1>
1763 0000B94E BE[5C940100]        <1>       mov   esi, mkdir_Name
1764 0000B953 66C7062E00          <1>       mov   word [esi], 2Eh ; db '.', '0'
1765                              <1>
1766 0000B958 A1[4C940100]        <1>       mov   eax, [mkdir_FFCluster]
1767 0000B95D 66B91000            <1>       mov   cx, 10h ; CL = Directory attribute
1768                              <1>                     ; CH = 0 -> File size is 0
1769 0000B961 E883000000          <1>       call  make_directory_entry
1770                              <1>
1771 0000B966 BF20000800          <1>       mov   edi, Directory_Buffer + 32 ; 26/02/2016
1772                              <1>
1773                              <1>       ; 03/03/2016
1774                              <1>       ; Following modification has been done according to
1775                              <1>       ; 'Microsoft Extensible Firmware Initiative
1776                              <1>       ; FAT32 File System Specification' document,
1777                              <1>       ; 'FAT: General Overview of On-Disk Format—Page 25'.
1778                              <1>       ; "Finally, you set DIR_FstClusLO and DIR_FstClusHI
1779                              <1>       ; for the dotdot entry (the second entry) to the
1780                              <1>       ; first cluster number of the directory in which you
1781                              <1>       ; just created the directory (value is 0 if this directory
1782                              <1>       ; is the root directory even for FAT32 volumes)."
1783                              <1>       ; (Correctness of this modification has been verified
1784                              <1>       ;  by using Windows 98 'scandisk.exe'.)
1785                              <1>
1786 0000B96B 29C0                <1>       sub   eax, eax
1787 0000B96D 3805[DC8A0100]      <1>       cmp   byte [Current_Dir_Level], al ; 0
1788 0000B973 7605                <1>       jna   short loc_mkdir_set_ff_dir_entry_3
1789 0000B975 A1[D88A0100]        <1>       mov   eax, [Current_Dir_FCluster] ; parent dir
1790                              <1> loc_mkdir_set_ff_dir_entry_3:
1791 0000B97A 66C746012E00        <1>       mov   word [esi+1], 2Eh ; db '.', '0'
1792                              <1>
1793                              <1>       ;mov   cx, 10h
1794 0000B980 E864000000          <1>       call  make_directory_entry
1795                              <1>
1796 0000B985 5E                  <1>       pop   esi
1797                              <1>
1798 0000B986 C605[04920100]02    <1>       mov   byte [DirBuff_ValidData], 2
1799 0000B98D E834010000          <1>       call  save_directory_buffer
1800 0000B992 0F8373FFFFFF        <1>         jnc     loc_mkdir_update_sub_dir_cluster
1801                              <1>
1802                              <1> retn_make_fat_directory:
1803 0000B998 C3                  <1>       retn
```

```
                                <1>
                                <1> loc_mkdir_save_fat_buffer_3:
                                <1>       ; 01/03/2016
                                <1>       ; byte [FAT_BuffValidData] = 2
0000B999 E8BF190000             <1>       call   save_fat_buffer
0000B99E 0F821AFFFFFF           <1>         jc      loc_mkdir_save_fat_buffer_stc_retn
                                <1>
0000B9A4 803D[FA910100]01       <1>       cmp    byte [FAT_ClusterCounter], 1
0000B9AB 721B                   <1>       jb     short loc_mkdir_save_fat_buffer_4
                                <1>
                                <1>       ; ESI = Logical DOS Drive Description Table address
0000B9AD A1[FA910100]           <1>       mov    eax, [FAT_ClusterCounter]
0000B9B2 66BB01FF               <1>       mov    bx, 0FF01h ; add free clusters
0000B9B6 E8371A0000             <1>       call   calculate_fat_freespace
                                <1>
                                <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
                                <1>         ;jnz    short loc_mkdir_save_fat_buffer_4
                                <1>
                                <1>       ; ecx > 0 -> Recalculation is needed
0000B9BB 09C9                   <1>       or     ecx, ecx
0000B9BD 7409                   <1>         jz      short loc_mkdir_save_fat_buffer_4
                                <1>
0000B9BF 66BB00FF               <1>       mov    bx, 0FF00h ; recalculate free space
0000B9C3 E82A1A0000             <1>       call   calculate_fat_freespace
                                <1>
                                <1> loc_mkdir_save_fat_buffer_4:
0000B9C8 C605[5B940100]00       <1>       mov    byte [mkdir_add_new_cluster], 0
                                <1>
                                <1> loc_mkdir_upd_parent_dir_lmdt:
0000B9CF E88D010000             <1>       call   update_parent_dir_lmdt
                                <1>
                                <1>       ; 01/03/2016
0000B9D4 803D[5B940100]00       <1>       cmp    byte [mkdir_add_new_cluster], 0
0000B9DB 0F8723FEFFFF           <1>         ja      loc_mkdir_gffc_2
                                <1>
                                <1> loc_mkdir_retn_new_dir_cluster:
0000B9E1 A1[4C940100]           <1>       mov    eax, [mkdir_FFCluster]
0000B9E6 31D2                   <1>       xor    edx, edx
                                <1> loc_mkdir_retn:
0000B9E8 C3                     <1>       retn
                                <1>
                                <1> make_directory_entry:
                                <1>       ; 02/03/2016
                                <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 09/08/2010 (DIR.ASM, 'proc_make_directory_entry')
                                <1>       ; 17/07/2010
                                <1>       ; INPUT ->
                                <1>       ;      EDI = Directory Entry Address
                                <1>       ;      ESI = Dot File Name Location
                                <1>       ;      EAX = First Cluster
                                <1>       ;      File Size = 0 (Must be set later)
                                <1>       ;      CL = Attributes
                                <1>       ;      CH = 0 (File size = 0)
                                <1>       ;      (If CH>0, File size is in dword [EBX]) (*)
                                <1>       ; OUTPUT ->
                                <1>       ;      EDI = Directory Entry Address
                                <1>       ;      ESI = Dot File Name Location (Capitalized)
                                <1>       ;      If CH input = 0, File Size = 0
                                <1>       ;      Otherwise file size is as dword [EBX] (*)
                                <1>       ;      DX = Date, AX = Time in DOS Dir Entry format
                                <1>       ;      EBX = same
                                <1>       ;      ECX = same
                                <1>
0000B9E9 51                     <1>       push   ecx
                                <1>
0000B9EA 884F0B                 <1>       mov    [edi+11], cl ; Attributes
0000B9ED 6689471A               <1>       mov    [edi+26], ax ; FClusterLw, 26
0000B9F1 C1E810                 <1>       shr    eax, 16
0000B9F4 66894714               <1>       mov    [edi+20], ax ; FClusterHw, 20
0000B9F8 6631C0                 <1>       xor    ax, ax
0000B9FB 6689470C               <1>       mov    [edi+12], ax ; NTReserved, 12
                                <1>                           ; CrtTimeTenth, 13
0000B9FF 08ED                   <1>       or     ch, ch
0000BA01 7402                   <1>       jz     short loc_make_direntry_set_filesize
                                <1>
0000BA03 8B03                   <1>       mov    eax, [ebx]
                                <1>
                                <1> loc_make_direntry_set_filesize:
0000BA05 89471C                 <1>       mov    [edi+28], eax ; FileSize, 28
                                <1>
0000BA08 E88AFBFFFF             <1>       call   convert_file_name
                                <1>       ;EDI = Dir Entry Format File Name Location
                                <1>       ;ESI = Dot File Name Location (capitalized)
                                <1>
0000BA0D E816000000             <1>       call   convert_current_date_time
                                <1>       ; OUTPUT -> DX = Date in dos dir entry format
                                <1>         ;       AX = Time in dos dir entry format
0000BA12 6689470E               <1>       mov    [edi+14], ax ; CrtTime, 14
0000BA16 66895710               <1>       mov    [edi+16], dx ; CrtDate, 16
0000BA1A 66895712               <1>       mov    [edi+18], dx ; LastAccDate, 18
0000BA1E 66894716               <1>       mov    [edi+22], ax ; WrtTime, 14
0000BA22 66895718               <1>       mov    [edi+24], dx ; WrtDate, 16
0000BA26 59                     <1>       pop    ecx
                                <1>
0000BA27 C3                     <1>       retn
                                <1>
                                <1> convert_current_date_time:
                                <1>       ; 21/02/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ; 13/06/2010 (DIR.ASM, 'proc_convert_current_date_time')
                                <1>       ; converts date&time to dos dir entry format
                                <1>       ; INPUT -> none
                                <1>       ; OUTPUT -> DX = Date in dos dir entry format
                                <1>       ;           AX = Time in dos dir entry format
                                <1>
0000BA28 B404                   <1>       mov    ah, 04h ; Return Current Date
```

```
1909 0000BA2A E80FB0FFFF          <1>       call  int1Ah
1910                              <1>
1911 0000BA2F 88E8                <1>       mov   al, ch ; <- century BCD
1912 0000BA31 240F                <1>       and   al, 0Fh
1913 0000BA33 88EC                <1>       mov   ah, ch
1914 0000BA35 C0EC04              <1>       shr   ah, 4
1915 0000BA38 D50A                <1>       aad
1916 0000BA3A 88C5                <1>       mov   ch, al ; -> century
1917                              <1>
1918 0000BA3C 88C8                <1>       mov   al, cl ; <- year BCD
1919 0000BA3E 240F                <1>       and   al, 0Fh
1920 0000BA40 88CC                <1>       mov   ah, cl
1921 0000BA42 C0EC04              <1>       shr   ah, 4
1922 0000BA45 D50A                <1>       aad
1923 0000BA47 88C1                <1>       mov   cl, al ; -> year
1924                              <1>
1925 0000BA49 88E8                <1>       mov   al, ch
1926 0000BA4B B464                <1>       mov   ah, 100
1927 0000BA4D F6E4                <1>       mul   ah
1928 0000BA4F 30ED                <1>       xor   ch, ch
1929 0000BA51 6601C8              <1>       add   ax, cx
1930 0000BA54 662DBC07            <1>       sub   ax, 1980 ; ms-dos epoch
1931 0000BA58 6689C1              <1>       mov   cx, ax
1932                              <1>
1933 0000BA5B 88F0                <1>       mov   al, dh ; <- month in bcd
1934 0000BA5D 240F                <1>       and   al, 0Fh
1935 0000BA5F 88F4                <1>       mov   ah, dh
1936 0000BA61 C0EC04              <1>       shr   ah, 4
1937 0000BA64 D50A                <1>       aad
1938 0000BA66 88C6                <1>       mov   dh, al ; -> month
1939                              <1>
1940 0000BA68 88D0                <1>       mov   al, dl ; <- day BCD
1941 0000BA6A 240F                <1>       and   al, 0Fh
1942 0000BA6C 88D4                <1>       mov   ah, dl
1943 0000BA6E C0EC04              <1>       shr   ah, 4
1944 0000BA71 D50A                <1>       aad
1945 0000BA73 88C2                <1>       mov   dl, al ; -> day
1946                              <1>
1947 0000BA75 88C8                <1>       mov   al, cl ; count of years from 1980
1948 0000BA77 66C1E004            <1>       shl   ax, 4
1949 0000BA7B 08F0                <1>       or    al, dh ; month of year, 1 to 12
1950 0000BA7D 66C1E005            <1>       shl   ax, 5
1951 0000BA81 08D0                <1>       or    al, dl ; day of year, 1 to 31
1952                              <1>
1953 0000BA83 6650                <1>       push  ax ; push date
1954                              <1>
1955 0000BA85 B402                <1>       mov   ah, 02h ; Return Current Time
1956 0000BA87 E8B2AFFFFF          <1>       call  int1Ah
1957                              <1>
1958 0000BA8C 88E8                <1>       mov   al, ch ; <- hours BCD
1959 0000BA8E 240F                <1>       and   al, 0Fh
1960 0000BA90 88EC                <1>       mov   ah, ch
1961 0000BA92 C0EC04              <1>       shr   ah, 4
1962 0000BA95 D50A                <1>       aad
1963 0000BA97 88C5                <1>       mov   ch, al ; -> hours
1964                              <1>
1965 0000BA99 88C8                <1>       mov   al, cl ; <- minutes BCD
1966 0000BA9B 240F                <1>       and   al, 0Fh
1967 0000BA9D 88CC                <1>       mov   ah, cl
1968 0000BA9F C0EC04              <1>       shr   ah, 4
1969 0000BAA2 D50A                <1>       aad
1970 0000BAA4 88C1                <1>       mov   cl, al ; -> minutes
1971                              <1>
1972 0000BAA6 88F0                <1>       mov   al, dh ; <- seconds BCD
1973 0000BAA8 240F                <1>       and   al, 0Fh
1974 0000BAAA 88F4                <1>       mov   ah, dh
1975 0000BAAC C0EC04              <1>       shr   ah, 4
1976 0000BAAF D50A                <1>       aad
1977 0000BAB1 88C6                <1>       mov   dh, al ; -> seconds
1978                              <1>
1979 0000BAB3 88E8                <1>       mov   al, ch ; hours
1980 0000BAB5 66C1E006            <1>       shl   ax, 6
1981 0000BAB9 08C8                <1>       or    al, cl ; minutes
1982 0000BABB 66C1E005            <1>       shl   ax, 5
1983 0000BABF D0EE                <1>       shr   dh, 1 ; 2 seconds
1984                              <1>       ; There is a bug in TRDOS v1 here !
1985                              <1>       ; it was 'or al, dl' !
1986 0000BAC1 08F0                <1>       or    al, dh ; seconds
1987                              <1>
1988 0000BAC3 665A                <1>       pop   dx ; pop date
1989                              <1>
1990 0000BAC5 C3                  <1>       retn
1991                              <1>
1992                              <1> save_directory_buffer:
1993                              <1>       ; 15/10/2016
1994                              <1>       ; 23/03/2016
1995                              <1>       ; 26/02/2016
1996                              <1>       ; 22/02/2016 (TRDOS 386 = TRDOS v2.0)
1997                              <1>       ; 01/08/2011
1998                              <1>       ; 14/03/2010
1999                              <1>       ; INPUT ->
2000                              <1>       ;       none
2001                              <1>       ; OUTPUT ->
2002                              <1>       ;  cf = 0 -> write OK...
2003                              <1>       ;  cf = 1 -> error code in AL (EAX)
2004                              <1>       ;  cf = 1 & AL = 0Dh => CH & CL = FS & FAT type
2005                              <1>       ;  EBX = Directory Buffer Address
2006                              <1>       ;
2007                              <1>       ;  (EAX, ECX, EDX will be modified)
2008                              <1>
2009 0000BAC6 BB00000800          <1>       mov   ebx, Directory_Buffer
2010 0000BACB 803D[04920100]02    <1>       cmp   byte [DirBuff_ValidData], 2
2011 0000BAD2 7403                <1>       je    short loc_save_dir_buffer
2012 0000BAD4 31C0                <1>       xor   eax, eax
2013 0000BAD6 C3                  <1>       retn
```

```
                                <1>
                                <1> loc_save_dir_buffer:
0000BAD7 56                     <1>       push   esi
0000BAD8 31DB                   <1>       xor    ebx, ebx
0000BADA 8A3D[02920100]         <1>         mov     bh, [DirBuff_DRV]
0000BAE0 80EF41                 <1>       sub    bh, 'A'
0000BAE3 BE00010900             <1>         mov     esi, Logical_DOSDisks
0000BAE8 01DE                   <1>       add    esi, ebx
0000BAEA 668B4E03               <1>         mov     cx, [esi+LD_FATType]
                                <1>       ; CH = FS Type (A1h for FS)
                                <1>       ; CL = FAT Type (0 for FS)
0000BAEE 08C9                   <1>       or     cl, cl
0000BAF0 7433                   <1>       jz     short loc_save_dir_buff_stc_retn
                                <1>
                                <1> loc_save_dir_buffer_check_cluster_no:
0000BAF2 A1[09920100]           <1>       mov    eax, [DirBuff_Cluster]
0000BAF7 28FF                   <1>       sub    bh, bh ; ebx = 0
0000BAF9 09C0                   <1>       or     eax, eax
0000BAFB 7540                   <1>       jnz    short loc_save_sub_dir_buffer
0000BAFD 8A25[03920100]         <1>       mov    ah, [DirBuff_FATType]
0000BB03 FEC3                   <1>       inc    bl ;  bl = 1
0000BB05 38DC                   <1>       cmp    ah, bl
0000BB07 721D                   <1>       jb     short loc_save_dir_buff_inv_data_retn
0000BB09 FEC3                   <1>       inc    bl ; bl = 2
0000BB0B 38E3                   <1>       cmp    bl, ah
0000BB0D 7217                   <1>       jb     short loc_save_dir_buff_inv_data_retn
                                <1>
                                <1> loc_save_root_dir_buffer:
0000BB0F 668B5E17               <1>       mov    bx, [esi+LD_BPB+RootDirEnts]
0000BB13 6683C30F               <1>       add    bx, 15
0000BB17 66C1EB04               <1>       shr    bx, 4 ; 16 dir entries per sector
0000BB1B 6609DB                 <1>       or     bx, bx
0000BB1E 7405                   <1>       jz     short loc_save_dir_buff_stc_retn
                                <1>       ;mov   ecx, ebx
0000BB20 8B4664                 <1>       mov    eax, [esi+LD_ROOTBegin] ; 26/02/2016
0000BB23 EB23                   <1>       jmp    short loc_write_directory_to_disk
                                <1>
                                <1> loc_save_dir_buff_stc_retn:
0000BB25 F9                     <1>       stc
                                <1> loc_save_dir_buff_inv_data_retn:
                                <1>       ; 15/10/2016 (0Dh -> 29)
0000BB26 B01D                   <1>       mov    al, 29 ; Invalid data !
0000BB28 C605[04920100]00       <1>       mov    byte [DirBuff_ValidData], 0
0000BB2F 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)
0000BB31 B812000000             <1>       mov    eax, 18 ; Drive not ready or write error
                                <1>
                                <1> loc_save_dir_buff_retn:
0000BB36 BB00000800             <1>       mov    ebx, Directory_Buffer
0000BB3B 5E                     <1>       pop    esi
0000BB3C C3                     <1>       retn
                                <1>
                                <1> loc_save_sub_dir_buffer:
                                <1>       ; ebx  = 0
0000BB3D 83E802                 <1>       sub    eax, 2
0000BB40 8A5E13                 <1>       mov    bl, [esi+LD_BPB+SecPerClust]
0000BB43 F7E3                   <1>       mul    ebx
0000BB45 034668                 <1>         add     eax, [esi+LD_DATABegin]
                                <1>       ;mov   ecx, ebx
                                <1>
                                <1> loc_write_directory_to_disk:
0000BB48 89D9                   <1>       mov    ecx, ebx
0000BB4A BB00000800             <1>       mov    ebx, Directory_Buffer
0000BB4F E83D700000             <1>       call   disk_write
0000BB54 72DB                   <1>       jc     short loc_write_directory_to_disk_err
                                <1>
                                <1> loc_save_dir_buff_validate_retn:
0000BB56 C605[04920100]01       <1>       mov    byte [DirBuff_ValidData], 1
0000BB5D 31C0                   <1>       xor    eax, eax
                                <1>       ; 26/02/2016
0000BB5F 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>
0000BB61 29C0                   <1>       sub    eax, eax
0000BB63 8A25[DC8A0100]         <1>       mov    ah, [Current_Dir_Level]
0000BB69 A0[DD8A0100]           <1>       mov    al, [Current_FATType]
0000BB6E 3C01                   <1>       cmp    al, 1
0000BB70 723A                   <1>       jb     short loc_UPDLMDT_proc_retn
                                <1>
                                <1> loc_update_parent_dir_lm_date_time:
0000BB72 08E4                   <1>       or     ah, ah
0000BB74 7436                   <1>       jz     short loc_UPDLMDT_proc_retn
                                <1>
0000BB76 56                     <1>       push   esi ; *
0000BB77 8825[7C940100]         <1>       mov    [UPDLMDT_CDirLevel], ah
0000BB7D 8B15[D88A0100]         <1>       mov    edx, [Current_Dir_FCluster]
0000BB83 8915[7D940100]         <1>       mov    [UPDLMDT_CDirFCluster], edx
                                <1>
0000BB89 FECC                   <1>       dec    ah
0000BB8B B90C000000             <1>       mov    ecx, 12
```

```
2119 0000BB90 BE[3B920100]        <1>         mov     esi, PATH_Array
2120                              <1>
2121 0000BB95 8825[DC8A0100]      <1>       mov   [Current_Dir_Level], ah
2122 0000BB9B 08E4                <1>       or    ah, ah
2123 0000BB9D 750E                <1>       jnz   short loc_update_parent_dir_lmdt_load_sub_dir_1
2124 0000BB9F 803D[DD8A0100]02    <1>       cmp   byte [Current_FATType], 2
2125 0000BBA6 770B                <1>       ja    short loc_update_parent_dir_lmdt_load_sub_dir_2
2126 0000BBA8 28C0                <1>       sub   al, al ; eax = 0
2127 0000BBAA EB0A                <1>       jmp   short loc_update_parent_dir_lmdt_load_sub_dir_3
2128                              <1>
2129                              <1> loc_UPDLMDT_proc_retn:
2130 0000BBAC C3                  <1>       retn
2131                              <1>
2132                              <1> loc_update_parent_dir_lmdt_load_sub_dir_1:
2133 0000BBAD B010                <1>       mov   al, 16
2134 0000BBAF F6E4                <1>       mul   ah
2135 0000BBB1 01C6                <1>       add   esi, eax
2136                              <1>
2137                              <1> loc_update_parent_dir_lmdt_load_sub_dir_2:
2138 0000BBB3 8B460C              <1>       mov   eax, [esi+12] ; Parent Dir First Cluster
2139                              <1>
2140                              <1> loc_update_parent_dir_lmdt_load_sub_dir_3:
2141 0000BBB6 A3[D88A0100]        <1>       mov   [Current_Dir_FCluster], eax
2142                              <1>
2143 0000BBBB 83C610              <1>       add   esi, 16
2144 0000BBBE 66BF[6293]          <1>       mov   di, Dir_File_Name
2145 0000BBC2 F3A4                <1>       rep   movsb
2146                              <1>
2147 0000BBC4 BE00010900          <1>       mov   esi, Logical_DOSDisks
2148 0000BBC9 29DB                <1>       sub   ebx, ebx
2149 0000BBCB 8A3D[DE8A0100]      <1>       mov   bh, [Current_Drv]
2150 0000BBD1 01DE                <1>       add   esi, ebx
2151 0000BBD3 E88EF7FFFF          <1>       call  reload_current_directory
2152 0000BBD8 7230                <1>       jc    short loc_update_parent_dir_lmdt_restore_cdirlevel
2153                              <1>
2154                              <1> loc_update_parent_dir_lmdt_locate_dir:
2155 0000BBDA BE[62930100]        <1>       mov   esi, Dir_File_Name
2156 0000BBDF 6631C9              <1>       xor   cx, cx
2157 0000BBE2 66B81008            <1>       mov   ax, 0810h ; Only directories
2158 0000BBE6 E8B4F6FFFF          <1>         call    locate_current_dir_file
2159                              <1>       ; EDI = DirBuff Directory Entry Address
2160 0000BBEB 721D                <1>       jc    short loc_update_parent_dir_lmdt_restore_cdirlevel
2161                              <1>
2162 0000BBED E836FEFFFF          <1>       call  convert_current_date_time
2163 0000BBF2 66895712            <1>       mov   [edi+18], dx ; Last Access Date
2164 0000BBF6 66895718            <1>       mov   [edi+24], dx ; Last Write Date
2165 0000BBFA 66894716            <1>       mov   [edi+22], ax ; Last Write Time
2166                              <1>
2167 0000BBFE C605[04920100]02    <1>       mov   byte [DirBuff_ValidData], 2
2168 0000BC05 E8BCFEFFFF          <1>       call  save_directory_buffer
2169                              <1>       ; 29/12/2017
2170                              <1>       ;jc   short loc_update_parent_dir_lmdt_restore_cdirlevel
2171                              <1>       ;xor  al, al
2172                              <1> loc_update_parent_dir_lmdt_restore_cdirlevel:
2173                              <1>       ;current directory level restoration
2174 0000BC0A 8A25[7C940100]      <1>       mov   ah, [UPDLMDT_CDirLevel]
2175 0000BC10 8825[DC8A0100]      <1>       mov   [Current_Dir_Level], ah
2176 0000BC16 8B15[7D940100]      <1>         mov     edx, [UPDLMDT_CDirFCluster]
2177 0000BC1C 8915[D88A0100]      <1>       mov   [Current_Dir_FCluster], edx
2178                              <1>
2179 0000BC22 5E                  <1>       pop   esi ; *
2180 0000BC23 C3                  <1>       retn
2181                              <1>
2182                              <1> delete_longname:
2183                              <1>       ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
2184                              <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_longname')
2185                              <1>       ; 14/03/2010
2186                              <1>       ; INPUT ->
2187                              <1>       ;     EAX = Directory Entry (Index) Number (< 65536)
2188                              <1>       ; OUTPUT ->
2189                              <1>       ;     cf = 0 -> OK  (EAX = 0)
2190                              <1>       ;     cf = 1 -> error code in EAX (AL)
2191                              <1>       ;
2192                              <1>       ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
2193                              <1>
2194 0000BC24 66A3[AC940100]      <1>       mov   [DLN_EntryNumber], ax
2195 0000BC2A C605[AE940100]40    <1>         mov     byte [DLN_40h], 40h
2196                              <1>
2197 0000BC31 E858000000          <1>       call  locate_current_dir_entry
2198 0000BC36 7308                <1>       jnc   short loc_dln_check_attributes
2199 0000BC38 C3                  <1>       retn
2200                              <1>
2201                              <1> loc_dln_longname_not_found:
2202 0000BC39 B802000000          <1>       mov   eax, 2
2203 0000BC3E F9                  <1>       stc
2204 0000BC3F C3                  <1>       retn
2205                              <1>
2206                              <1> loc_dln_check_attributes:
2207 0000BC40 B00F                <1>       mov   al, 0Fh  ; long name
2208 0000BC42 8A670B              <1>       mov   ah, [edi+0Bh] ; dir entry attributes
2209 0000BC45 38C4                <1>       cmp   ah, al
2210 0000BC47 75F0                <1>       jne   short loc_dln_longname_not_found
2211 0000BC49 8A27                <1>       mov   ah, [edi]
2212 0000BC4B 2A25[AE940100]      <1>       sub   ah, [DLN_40h]
2213 0000BC51 76E6                <1>       jna   short loc_dln_longname_not_found
2214 0000BC53 80FC14              <1>       cmp   ah, 14h ; 84-64=20 -> 20*13=260 bytes
2215 0000BC56 77E1                <1>       ja    short loc_dln_longname_not_found
2216                              <1>
2217 0000BC58 C607E5              <1>       mov   byte [edi], 0E5h  ; deleted sign
2218 0000BC5B C605[04920100]02    <1>       mov   byte [DirBuff_ValidData], 2 ; changed/write sign
2219 0000BC62 C605[AE940100]00    <1>       mov   byte [DLN_40h], 0 ; 40h -> 0
2220                              <1>
2221                              <1> loc_dln_delete_next_ln_entry:
2222 0000BC69 80FC01              <1>       cmp   ah, 1
2223 0000BC6C 7616                <1>       jna   short loc_dln_longname_retn
```

```
2224                                <1> loc_dln_delete_next_ln_entry_0:
2225 0000BC6E 66FF05[AC940100]     <1>         inc     word [DLN_EntryNumber]
2226 0000BC75 0FB705[AC940100]     <1>         movzx   eax, word [DLN_EntryNumber]
2227 0000BC7C E80D000000           <1>         call    locate_current_dir_entry
2228 0000BC81 73BD                 <1>         jnc     short loc_dln_check_attributes
2229                                <1>
2230                                <1> loc_dln_longname_stc_retn:
2231 0000BC83 C3                   <1>         retn
2232                                <1>
2233                                <1> loc_dln_longname_retn:
2234                                <1>         ;cmp    byte [DirBuff_ValidData], 2
2235                                <1>         ;jne    short loc_dln_longname_retn_xor_eax
2236 0000BC84 E83DFEFFFF           <1>         call    save_directory_buffer
2237 0000BC89 72F8                 <1>         jc      short loc_dln_longname_stc_retn
2238                                <1>
2239                                <1> loc_dln_longname_retn_xor_eax:
2240 0000BC8B 31C0                 <1>         xor     eax, eax
2241 0000BC8D C3                   <1>         retn
2242                                <1>
2243                                <1> locate_current_dir_entry:
2244                                <1>         ; 16/10/2016
2245                                <1>         ; 15/10/2016
2246                                <1>         ; 23/03/2016
2247                                <1>         ; 27/02/2016 (TRDOS 386 = TRDOS v2.0)
2248                                <1>         ; 01/08/2011 (DIR.ASM, 'proc_locate_current_dir_entry')
2249                                <1>         ; 07/03/2010
2250                                <1>         ; INPUT ->
2251                                <1>         ;       EAX = Directory Entry (Index) Number (< 65536)
2252                                <1>         ; OUTPUT ->
2253                                <1>         ;       EDI = Directory Entry Address
2254                                <1>         ;       EAX = Cluster Number of Directory Buffer
2255                                <1>         ;       EBX = Directory Buffer Entry Offset
2256                                <1>         ;       ECX = DirBuff Valid Data identifier (CL)
2257                                <1>         ;       If CF = 0 and CL = 2 then
2258                                <1>         ;          directory buffer modified and
2259                                <1>         ;          must be written to disk.
2260                                <1>         ;       If CF = 0  and CL = 1 then
2261                                <1>         ;          dir buffer has been written to disk, already.
2262                                <1>         ;       CF = 1 -> Error code in EAX (AL)
2263                                <1>         ;
2264                                <1>         ; (Modified registers: EAX, EDX, ECX, EBX, EDI)
2265                                <1>
2266                                <1> loc_locate_current_dir_entry:
2267 0000BC8E 56                   <1>         push    esi
2268 0000BC8F 89C1                 <1>         mov     ecx, eax
2269 0000BC91 BA20000000           <1>         mov     edx, 32
2270 0000BC96 F7E2                 <1>         mul     edx
2271 0000BC98 A3[B8940100]         <1>         mov     [LCDE_ByteOffset], eax
2272 0000BC9D 31DB                 <1>         xor     ebx, ebx
2273 0000BC9F 8A3D[DE8A0100]       <1>         mov     bh, [Current_Drv]
2274 0000BCA5 A0[02920100]         <1>           mov     al, [DirBuff_DRV]
2275 0000BCAA 2C41                 <1>         sub     al, 'A'
2276 0000BCAC BE00010900           <1>           mov     esi, Logical_DOSDisks
2277 0000BCB1 01DE                 <1>         add     esi, ebx
2278 0000BCB3 38C7                 <1>         cmp     bh, al
2279 0000BCB5 0F8592000000         <1>           jne     loc_lcde_reload_current_directory
2280                                <1> loc_lcde_cdl_check:
2281 0000BCBB 803D[DC8A0100]00     <1>         cmp     byte [Current_Dir_Level], 0
2282 0000BCC2 772A                 <1>         ja      short loc_lcde_calc_dirbuff_cluster_offset
2283                                <1>         ; 27/02/2016
2284                                <1>         ; TRDOS v1 has bug here for FAT32 fs !
2285                                <1>         ; (Root Directory Entries for FAT32 = 0)
2286 0000BCC4 807E0303             <1>         cmp     byte [esi+LD_FATType], 3  ; FAT32
2287 0000BCC8 7324                 <1>         jnb     short loc_lcde_calc_dirbuff_cluster_offset
2288                                <1>
2289                                <1> loc_lcde_cdl_check_FAT12_16:
2290 0000BCCA 668B4617             <1>         mov     ax, [esi+LD_BPB+RootDirEnts]
2291 0000BCCE 6648                 <1>         dec     ax
2292                                <1>         ;xor    dx, dx
2293 0000BCD0 6639C8               <1>         cmp     ax, cx ; cx = Directory Entry (Index) Number
2294 0000BCD3 720E                 <1>         jb      short loc_lcde_stc_12h_retn
2295 0000BCD5 66890D[B0940100]     <1>         mov     [LCDE_EntryIndex], cx
2296 0000BCDC 31C0                 <1>         xor     eax, eax
2297 0000BCDE E993000000           <1>           jmp     loc_lcde_check_dir_buffer_cluster
2298                                <1>
2299                                <1> loc_lcde_stc_12h_retn:
2300 0000BCE3 5E                   <1>         pop     esi
2301 0000BCE4 89CB                 <1>         mov     ebx, ecx
2302 0000BCE6 89D1                 <1>         mov     ecx, edx
2303                                <1>         ; 16/10/2016 (12h -> 12)
2304 0000BCE8 B80C000000           <1>         mov     eax, 12 ; No more files
2305 0000BCED C3                   <1>         retn
2306                                <1>
2307                                <1> loc_lcde_calc_dirbuff_cluster_offset:
2308 0000BCEE 8A5E13               <1>         mov     bl, [esi+LD_BPB+SecPerClust]
2309 0000BCF1 30FF                 <1>         xor     bh, bh
2310 0000BCF3 668B4611             <1>         mov     ax, [esi+LD_BPB+BytesPerSec]
2311 0000BCF7 66F7E3               <1>         mul     bx
2312 0000BCFA 6609D2               <1>         or      dx, dx ; If bytes per cluster > 32KB it is invalid
2313 0000BCFD 755D                 <1>         jnz     short loc_lcde_invalid_format
2314                                <1>         ;mov    ecx, eax
2315 0000BCFF 6689C1               <1>         mov     cx, ax ; BYTES PER CLUSTER
2316 0000BD02 A1[B8940100]         <1>         mov     eax, [LCDE_ByteOffset]
2317                                <1>         ;sub    edx, edx
2318 0000BD07 F7F1                 <1>         div     ecx
2319 0000BD09 3DFFFF0000           <1>         cmp     eax, 65535
2320 0000BD0E 774C                 <1>         ja      short loc_lcde_invalid_format
2321                                <1>
2322                                <1>         ; cluster sequence number of directory (< 65536)
2323 0000BD10 66A3[B2940100]       <1>         mov     [LCDE_ClusterSN], ax
2324                                <1>
2325 0000BD16 6689D0               <1>         mov     ax, dx ; byte offset in cluster (directory buffer)
2326 0000BD19 66BB2000             <1>         mov     bx, 32 ; ; 1 dir entry = 32 bytes
2327 0000BD1D 6629D2               <1>           sub     dx, dx  ; 0
2328 0000BD20 66F7F3               <1>         div     bx
```

```
2329 0000BD23 66A3[B0940100]      <1>       mov    [LCDE_EntryIndex], ax ; dir entry index/sequence number
2330                              <1>                                    ; (in directory buffer/cluster)
2331                              <1> loc_lcde_get_current_sub_dir_fcluster:
2332 0000BD29 A1[D88A0100]        <1>       mov    eax, [Current_Dir_FCluster]
2333                              <1>
2334                              <1> loc_lcde_get_next_cluster:
2335 0000BD2E 66833D[B2940100]00  <1>       cmp    word [LCDE_ClusterSN], 0
2336 0000BD36 763E                <1>       jna    short loc_lcde_check_dir_buffer_cluster
2337 0000BD38 A3[B4940100]        <1>       mov    [LCDE_Cluster], eax
2338 0000BD3D E834100000          <1>       call   get_next_cluster
2339 0000BD42 7220                <1>       jc     short loc_lcde_check_gnc_error
2340 0000BD44 66FF0D[B2940100]    <1>       dec    word [LCDE_ClusterSN]
2341 0000BD4B EBE1                <1>       jmp    short loc_lcde_get_next_cluster
2342                              <1>
2343                              <1> loc_lcde_reload_current_directory:
2344 0000BD4D 51                  <1>       push   ecx
2345 0000BD4E E813F6FFFF          <1>       call   reload_current_directory
2346 0000BD53 59                  <1>       pop    ecx
2347 0000BD54 0F8361FFFFFF        <1>         jnc     loc_lcde_cdl_check
2348 0000BD5A 5E                  <1>       pop    esi
2349 0000BD5B C3                  <1>       retn
2350                              <1>
2351                              <1> loc_lcde_invalid_format:
2352                              <1>       ; 15/10/2016 (0Bh -> 28)
2353 0000BD5C B81C000000          <1>       mov    eax, 28 ; Invalid Format !
2354                              <1> loc_lcde_drive_not_ready_read_err:
2355 0000BD61 F9                  <1>       stc
2356 0000BD62 5E                  <1>       pop    esi
2357 0000BD63 C3                  <1>       retn
2358                              <1>
2359                              <1> loc_lcde_check_gnc_error:
2360 0000BD64 09C0                <1>       or     eax, eax
2361 0000BD66 75F9                <1>       jnz    short loc_lcde_drive_not_ready_read_err
2362 0000BD68 66FF0D[B2940100]    <1>       dec    word [LCDE_ClusterSN]
2363 0000BD6F 75EB                <1>       jnz    short loc_lcde_invalid_format
2364 0000BD71 A1[B4940100]        <1>       mov    eax, [LCDE_Cluster]
2365                              <1>
2366                              <1> loc_lcde_check_dir_buffer_cluster:
2367 0000BD76 3B05[09920100]      <1>       cmp    eax, [DirBuff_Cluster]
2368 0000BD7C 755C                <1>       jne    short loc_lcde_load_dir_cluster
2369 0000BD7E 803D[04920100]00    <1>       cmp    byte [DirBuff_ValidData], 0
2370 0000BD85 7727                <1>       ja     short lcde_check_dir_buffer_cluster_next
2371 0000BD87 803D[DC8A0100]00    <1>       cmp    byte [Current_Dir_Level], 0
2372 0000BD8E 775F                <1>       ja     short loc_lcde_load_dir_cluster_0
2373                              <1>       ; 27/02/2016
2374                              <1>       ; TRDOS v1 has bug here for FAT32 fs !
2375 0000BD90 807E0303            <1>       cmp    byte [esi+LD_FATType], 3  ; FAT32
2376 0000BD94 7359                <1>       jnb    short loc_lcde_load_dir_cluster_0
2377                              <1>       ;
2378 0000BD96 0FB74E17            <1>       movzx  ecx, word [esi+LD_BPB+RootDirEnts]
2379 0000BD9A 6683C10F            <1>       add    cx, 15 ; round up (16 entries per sector)
2380 0000BD9E 66C1E904            <1>       shr    cx, 4 ; 1 sector contains 16 dir entries
2381                              <1>
2382 0000BDA2 8B4664              <1>         mov     eax, [esi+LD_ROOTBegin]
2383 0000BDA5 EB54                <1>       jmp    short loc_lcde_load_dir_cluster_1
2384                              <1>
2385                              <1> loc_lcde_validate_dirBuff:
2386 0000BDA7 C605[04920100]01    <1>       mov    byte [DirBuff_ValidData], 1
2387                              <1>
2388                              <1> lcde_check_dir_buffer_cluster_next:
2389 0000BDAE 0FB71D[B0940100]    <1>       movzx  ebx, word [LCDE_EntryIndex]
2390 0000BDB5 663B1D[07920100]    <1>       cmp    bx, [DirBuff_LastEntry]
2391 0000BDBC 779E                <1>       ja     short loc_lcde_invalid_format
2392 0000BDBE B820000000          <1>       mov    eax, 32
2393 0000BDC3 F7E3                <1>       mul    ebx
2394                              <1>       ;or    edx, edx
2395                              <1>       ;jnz   short loc_lcde_invalid_format
2396                              <1>
2397 0000BDC5 BF00000800          <1>       mov    edi, Directory_Buffer
2398 0000BDCA 01C7                <1>       add    edi, eax ; add entry offset to buffer address
2399                              <1>
2400                              <1> loc_lcde_dir_buffer_last_check:
2401 0000BDCC A1[09920100]        <1>       mov    eax, [DirBuff_Cluster]
2402 0000BDD1 0FB60D[04920100]    <1>       movzx  ecx, byte [DirBuff_ValidData]
2403                              <1>
2404                              <1> loc_lcde_retn:
2405 0000BDD8 5E                  <1>       pop    esi
2406 0000BDD9 C3                  <1>       retn
2407                              <1>
2408                              <1> loc_lcde_load_dir_cluster:
2409                              <1>       ;cmp   byte [DirBuff_ValidData], 2
2410                              <1>       ;jne   short loc_lcde_load_dir_cluster_n2
2411 0000BDDA 50                  <1>       push   eax
2412 0000BDDB E8E6FCFFFF          <1>       call   save_directory_buffer
2413 0000BDE0 58                  <1>       pop    eax
2414 0000BDE1 72F5                <1>       jc     short loc_lcde_retn
2415                              <1>
2416                              <1> loc_lcde_load_dir_cluster_n2:
2417 0000BDE3 C605[04920100]00    <1>       mov    byte [DirBuff_ValidData], 0
2418 0000BDEA A3[09920100]        <1>       mov    [DirBuff_Cluster], eax
2419                              <1>
2420                              <1> loc_lcde_load_dir_cluster_0:
2421 0000BDEF 83E802              <1>       sub    eax, 2
2422 0000BDF2 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
2423 0000BDF6 F7E1                <1>       mul    ecx
2424 0000BDF8 034668              <1>         add     eax, [esi+LD_DATABegin]
2425                              <1>
2426                              <1> loc_lcde_load_dir_cluster_1:
2427 0000BDFB BB00000800          <1>       mov    ebx, Directory_Buffer
2428                              <1>       ; ecx = sector count
2429 0000BE00 E89B6D0000          <1>       call   disk_read
2430 0000BE05 73A0                <1>       jnc    short loc_lcde_validate_dirBuff
2431                              <1>
2432                              <1>       ; 15/10/2016
2433                              <1>       ; (Disk read error instead of drv not ready err)
```

```
0000BE07 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error !
0000BE0C EBCA                <1>       jmp    short loc_lcde_retn
                             <1>
                             <1>
                             <1> remove_file:
                             <1>       ; 15/10/2016
                             <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ; 10/04/2011 (FILE.ASM, 'proc_delete_file')
                             <1>       ; 09/08/2010
                             <1>       ; INPUT ->
                             <1>       ;      EDI = Directory Buffer Entry Address
                             <1>       ;       CX = Directory Buffer Entry Counter/Index
                             <1>       ;       BL = Longname Entry Length
                             <1>       ;       BH = Logical DOS Drive Number
                             <1>
0000BE0E 29C0                <1>       sub    eax, eax
0000BE10 88FC                <1>       mov    ah, bh
0000BE12 BE00010900          <1>       mov    esi, Logical_DOSDisks
0000BE17 01C6                <1>       add    esi, eax
                             <1>
0000BE19 807E0301            <1>       cmp    byte [esi+LD_FATType], 1
0000BE1D 7312                <1>       jnb    short loc_del_fat_file
                             <1>
0000BE1F 807E04A1            <1>       cmp    byte [esi+LD_FSType], 0A1h
0000BE23 7406                <1>       je     short loc_del_fs_file
                             <1>
                             <1> loc_del_file_invalid_format:
0000BE25 30E4                <1>       xor    ah, ah
                             <1>       ; 15/10/2016 (0Bh -> 28)
0000BE27 B01C                <1>       mov    al, 28  ; Invalid Format
0000BE29 F9                  <1>       stc
0000BE2A C3                  <1>       retn
                             <1>
                             <1> loc_del_fs_file:
0000BE2B E83F0F0000          <1>       call   delete_fs_file
0000BE30 C3                  <1>       retn
                             <1>
                             <1> loc_del_fat_file:
0000BE31 E808000000          <1>       call   delete_directory_entry
0000BE36 7205                <1>       jc     short loc_del_file_err_retn
                             <1>
                             <1> loc_delfile_unlink_cluster_chain:
0000BE38 E863170000          <1>       call   truncate_cluster_chain
                             <1>       ;jc    short loc_del_file_err_retn
                             <1>
                             <1> loc_delfile_return:
                             <1> loc_del_file_err_retn:
0000BE3D C3                  <1>       retn
                             <1>
                             <1> delete_directory_entry:
                             <1>       ; 15/10/2016
                             <1>       ; 28/02/2016 (TRDOS 386 = TRDOS v2.0)
                             <1>       ; 01/08/2011 (DIR.ASM, 'proc_delete_directory_entry')
                             <1>       ; 10/04/2011
                             <1>       ; INPUT ->
                             <1>       ;      ESI = Logical Dos Drive Descripton Table Address
                             <1>       ;      EDI = Directory Buffer Entry Address
                             <1>       ;       CX = Directory Buffer Entry Counter/Index
                             <1>       ;       BL = Longname Entry Length
                             <1>       ; OUTPUT ->
                             <1>       ;      ESI = Logical dos drive descripton table address
                             <1>       ;      EAX = First cluster to be truncated/unlinked
                             <1>       ;       CF = 1 -> Error code in EAX (AL)
                             <1>       ;       CF = 0 & BH <> 0 -> LMDT write error  (BH = 1)
                             <1>       ;       CF = 0 & BL <> 0 -> Long name delete error (BL = FFh)
                             <1>       ;
                             <1>       ;  (EDI, EBX, ECX register contents will be changed)
                             <1>
0000BE3E 881D[46940100]      <1>       mov    [DelFile_LNEL], bl
0000BE44 66890D[44940100]    <1>       mov    [DelFile_EntryCounter], cx
                             <1>
0000BE4B 668B4714            <1>       mov    ax, [edi+20] ; First Cluster High Word
0000BE4F C1E010              <1>       shl    eax, 16
0000BE52 668B471A            <1>       mov    ax, [edi+26] ; First Cluster Low Word
                             <1>
0000BE56 A3[40940100]        <1>       mov    [DelFile_FCluster], eax
                             <1>
                             <1> loc_del_short_name:
0000BE5B C607E5              <1>       mov    byte [edi], 0E5h  ; Deleted sign
                             <1>
0000BE5E C605[04920100]02    <1>       mov    byte [DirBuff_ValidData], 2
0000BE65 E85CFCFFFF          <1>       call   save_directory_buffer
0000BE6A 723D                <1>       jc     short loc_delete_direntry_err_return
                             <1>
                             <1> loc_del_long_name:
0000BE6C 0FB615[46940100]    <1>       movzx  edx, byte [DelFile_LNEL]
0000BE73 08D2                <1>       or     dl, dl
0000BE75 7416                <1>       jz     short loc_del_dir_entry_update_parent_dir_lm_date
                             <1>
0000BE77 8835[46940100]      <1>       mov    byte [DelFile_LNEL], dh ; 0
                             <1>
0000BE7D 0FB705[44940100]    <1>       movzx  eax,  word [DelFile_EntryCounter]
0000BE84 29D0                <1>       sub    eax, edx
                             <1>       ;jnc   short loc_del_long_name_continue
0000BE86 7205                <1>       jc     short loc_del_dir_entry_update_parent_dir_lm_date
                             <1>
                             <1> ;loc_del_direntry_inv_data_return: ; 15/10/2016 (0Dh -> 29)
                             <1> ;     mov    eax, 29 ; 0Dh (TRDOS 8086) ; Invalid data
                             <1> ;     retn
                             <1>
                             <1> loc_del_long_name_continue:
                             <1>       ; AX = Directory Entry Number of the long name last entry
0000BE88 E897FDFFFF          <1>       call   delete_longname
                             <1>       ;jc    short loc_delete_direntry_err_return
                             <1>
```

```
                            <1> loc_del_dir_entry_update_parent_dir_lm_date:
0000BE8D 801D[46940100]00   <1>       sbb    byte [DelFile_LNEL], 0 ; 0FFh if cf = 1
                            <1>
0000BE94 E8C8FCFFFF         <1>       call   update_parent_dir_lmdt
0000BE99 B700               <1>       mov    bh, 0
0000BE9B 80D700             <1>       adc    bh, 0
                            <1>
0000BE9E 8A1D[46940100]     <1>       mov    bl, byte [DelFile_LNEL]
                            <1>
                            <1> loc_delete_direntry_return:
0000BEA4 A1[40940100]       <1>       mov    eax, [DelFile_FCluster]
                            <1> loc_delete_direntry_err_return:
0000BEA9 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>
0000BEAA 803D[DD8A0100]00   <1>       cmp    byte [Current_FATType], 0
0000BEB1 7706               <1>       ja     short loc_rename_directory_entry
                            <1>
0000BEB3 E8B80E0000         <1>       call   rename_fs_file_or_directory
0000BEB8 C3                 <1>       retn
                            <1>
                            <1> loc_rename_directory_entry:
0000BEB9 881D[46940100]     <1>       mov    [DelFile_LNEL], bl
0000BEBF 66890D[44940100]   <1>       mov    [DelFile_EntryCounter], cx
0000BEC6 A3[40940100]       <1>       mov    [DelFile_FCluster], eax
                            <1>
0000BECB 0FB7C1             <1>       movzx  eax, cx
0000BECE E8BBFDFFFF         <1>       call   locate_current_dir_entry
0000BED3 7308               <1>       jnc    short loc_rename_direntry_check_fcluster
                            <1>
                            <1> loc_rename_direntry_pop_retn:
0000BED5 C3                 <1>       retn
                            <1>
                            <1> loc_rename_direntry_pop_invd_retn:
0000BED6 F9                 <1>       stc
                            <1> loc_rename_direntry_invd_retn:
                            <1>       ; 15/10/2016 (0Dh -> 29)
0000BED7 B81D000000         <1>       mov    eax, 29 ; Invalid data
                            <1> loc_rename_retn:
0000BEDC C3                 <1>       retn
                            <1>
                            <1> loc_rename_direntry_check_fcluster:
0000BEDD 668B5714           <1>       mov    dx, [edi+20] ; First Cluster HW
0000BEE1 C1E210             <1>       shl    edx, 16 ; 13/11/2017
0000BEE4 668B571A           <1>       mov    dx, [edi+26] ; First Cluster LW
0000BEE8 3B15[40940100]     <1>       cmp    edx, [DelFile_FCluster]
0000BEEE 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.)
0000BEF0 E8A2F6FFFF         <1>       call   convert_file_name
                            <1>
0000BEF5 C605[04920100]02   <1>         mov     byte [DirBuff_ValidData], 2
0000BEFC E8C5FBFFFF         <1>       call   save_directory_buffer
0000BF01 72D9               <1>       jc     short loc_rename_retn
                            <1>
                            <1> loc_rename_direntry_del_ln:
0000BF03 0FB615[46940100]   <1>       movzx  edx, byte [DelFile_LNEL]
0000BF0A 08D2               <1>       or     dl, dl
0000BF0C 7410               <1>       jz     short loc_rename_direntry_update_parent_dir_lm_date
                            <1>
0000BF0E 0FB705[44940100]   <1>       movzx  eax, word [DelFile_EntryCounter]
0000BF15 29D0               <1>       sub    eax, edx
0000BF17 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
0000BF19 E806FDFFFF         <1>       call   delete_longname
                            <1>
                            <1> loc_rename_direntry_update_parent_dir_lm_date:
0000BF1E E83EFCFFFF         <1>       call   update_parent_dir_lmdt
0000BF23 31C0               <1>       xor    eax, eax
0000BF25 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,
```

```
2644                                    <1>       ;              'not found' is required
2645                                    <1>       ;   Phase 2 -> Check source file
2646                                    <1>       ;              'found' and proper attributes is required
2647                                    <1>       ;   Phase 3 -> Make destination directory entry,
2648                                    <1>       ;             add new dir cluster or section if it is required
2649                                    <1>       ;   Phase 4 -> Delete source directory entry.
2650                                    <1>       ;       cf = 1 causes to return before the phase 4.
2651                                    <1>       ;    (source file protection against any possible errors)
2652                                    <1>       ;
2653                                    <1>       ; 08/05/2011 major modification
2654                                    <1>       ;            -> destination file deleting is removed
2655                                    <1>       ;            for msdos move/rename compatibility.
2656                                    <1>       ;            (Access denied error will return if
2657                                    <1>       ;            the destination file is found...)
2658                                    <1>       ; INPUT ->
2659                                    <1>       ;      ESI = Source File Pathname (Asciiz)
2660                                    <1>       ;        EDI = Destination File Pathname (Asciiz)
2661                                    <1>       ;        AL = 0 --> Interrupt (System call)
2662                                    <1>       ;        AL > 0 --> Command Interpreter (Question)
2663                                    <1>       ;        AL = 1 --> Question Phase
2664                                    <1>       ;        AL = 2 --> Progress Phase
2665                                    <1>       ; OUTPUT ->
2666                                    <1>       ;      cf = 0 -> OK
2667                                    <1>       ;        EAX = Destination directory first cluster
2668                                    <1>       ;        ESI = Logical DOS drive description table
2669                                    <1>       ;        EBX = Destination file structure offset
2670                                    <1>       ;        CX = 0 (CX > 0 --> calculate free space error)
2671                                    <1>       ;        cf = 1 -> Error code in EAX (AL)
2672                                    <1>       ;
2673                                    <1>       ;  (EDX, ECX, EBX, ESI, EDI will be changed)
2674                                    <1>
2675 0000BF26 3C02                     <1>       cmp    al, 2
2676 0000BF28 0F847F010000             <1>       je     msftdf_df2_check_directory
2677 0000BF2E A2[C6950100]             <1>       mov    [move_cmd_phase], al
2678                                    <1>
2679                                    <1> msftdf_parse_sf_path:
2680                                    <1>       ; ESI = ASCIIZ pathname (Source)
2681 0000BF33 57                       <1>       push   edi
2682 0000BF34 BF[C4940100]             <1>       mov    edi, SourceFile_Drv
2683 0000BF39 E824F7FFFF               <1>       call   parse_path_name
2684 0000BF3E 5E                       <1>       pop    esi
2685 0000BF3F 7211                     <1>       jc     short msftdf_psf_retn
2686                                    <1>
2687                                    <1> msftdf_parse_df_path:
2688                                    <1>       ; ESI = ASCIIZ pathname   (Destination)
2689 0000BF41 BF[44950100]             <1>       mov    edi, DestinationFile_Drv
2690 0000BF46 E817F7FFFF               <1>       call   parse_path_name
2691 0000BF4B 7306                     <1>       jnc    short msftdf_check_sf_drv
2692                                    <1>
2693 0000BF4D 3C01                     <1>       cmp    al, 1 ; File or directory name is not existing
2694 0000BF4F 7602                     <1>       jna    short msftdf_check_sf_drv
2695                                    <1>
2696                                    <1> msftdf_stc_retn:
2697 0000BF51 F9                       <1>       stc
2698                                    <1> msftdf_psf_retn:
2699 0000BF52 C3                       <1>       retn
2700                                    <1>
2701                                    <1> msftdf_check_sf_drv:
2702 0000BF53 A0[C4940100]             <1>       mov    al, [SourceFile_Drv]
2703                                    <1>
2704                                    <1> msftdf_check_df_drv:
2705 0000BF58 8A15[44950100]           <1>       mov    dl, [DestinationFile_Drv]
2706                                    <1>
2707                                    <1> msftdf_compare_sf_df_drv:
2708 0000BF5E 29DB                     <1>       sub    ebx, ebx
2709 0000BF60 8A3D[DE8A0100]           <1>       mov    bh, [Current_Drv]
2710 0000BF66 38C2                     <1>       cmp    dl, al
2711 0000BF68 7409                     <1>       je     short msftdf_check_sf_df_drv_ok
2712                                    <1>
2713                                    <1> msftdf_not_same_drv:
2714                                    <1>         ; DL = source file's drive number
2715 0000BF6A 88C6                     <1>       mov    dh, al ; destination file's drive number
2716                                    <1>       ; 15/10/2016 (11h -> 21)
2717 0000BF6C B815000000               <1>       mov    eax, 21 ; Not the same drive
2718 0000BF71 F9                       <1>       stc
2719 0000BF72 C3                       <1>       retn
2720                                    <1>
2721                                    <1> msftdf_check_sf_df_drv_ok:
2722 0000BF73 8815[C7950100]           <1>       mov    [msftdf_sf_df_drv], dl
2723                                    <1>
2724 0000BF79 29C0                     <1>         sub  eax, eax
2725 0000BF7B 88D4                     <1>       mov    ah, dl
2726 0000BF7D 0500010900               <1>       add    eax, Logical_DOSDisks
2727 0000BF82 A3[C8950100]             <1>       mov    [msftdf_drv_offset], eax
2728                                    <1>
2729 0000BF87 38FA                     <1>       cmp    dl, bh ; byte [Current_Drv]
2730 0000BF89 7407                     <1>       je     short msftdf_df_check_directory
2731                                    <1>
2732                                    <1> msftdf_change_drv:
2733 0000BF8B E85EC1FFFF               <1>       call   change_current_drive
2734 0000BF90 726D                     <1>       jc     short msftdf_df_error_retn
2735                                    <1>
2736                                    <1> msftdf_check_destination_file:
2737                                    <1> msftdf_df_check_directory:
2738 0000BF92 BE[45950100]             <1>       mov    esi, DestinationFile_Directory
2739 0000BF97 803E20                   <1>       cmp    byte [esi], 20h
2740 0000BF9A 760F                     <1>       jna    short msftdf_df_find_1
2741                                    <1>
2742                                    <1> msftdf_df_change_directory:
2743 0000BF9C FE05[A6400100]           <1>       inc    byte [Restore_CDIR]
2744 0000BFA2 30E4                     <1>       xor    ah, ah ; CD_COMMAND sign -> 0
2745 0000BFA4 E8A3F0FFFF               <1>       call   change_current_directory
2746 0000BFA9 7254                     <1>       jc     short msftdf_df_error_retn
2747                                    <1>
2748                                    <1> ;msftdf_df_change_prompt_dir_string:
```

```
2749                            <1> ;       call    change_prompt_dir_string
2750                            <1>
2751                            <1> msftdf_df_find_1:
2752 0000BFAB BE[86950100]      <1>         mov     esi, DestinationFile_Name
2753 0000BFB0 803E20            <1>         cmp     byte [esi], 20h
2754 0000BFB3 7631              <1>         jna     short msftdf_df_copy_sf_name
2755                            <1>
2756                            <1> msftdf_df_find_2:
2757 0000BFB5 6631C0            <1>         xor     ax, ax ; DestinationFile_AttributesMask -> any/zero
2758 0000BFB8 E8D4D4FFFF        <1>         call    find_first_file
2759 0000BFBD 0F838D000000      <1>         jnc     msftdf_permission_denied_retn
2760                            <1>
2761                            <1> msftdf_df_check_error_code:
2762                            <1>         ;cmp    eax, 2 ; File not found error
2763 0000BFC3 3C02              <1>         cmp     al, 2
2764 0000BFC5 7537              <1>         jne     short msftdf_df_stc_retn
2765                            <1>
2766                            <1> msftdf_df_check_fname:
2767                            <1>         ; 15/10/2016
2768 0000BFC7 BE[86950100]      <1>         mov     esi, DestinationFile_Name ; *
2769 0000BFCC E87ED8FFFF        <1>         call    check_filename
2770 0000BFD1 7307              <1>         jnc     short msftdf_convert_df_direntry_name
2771                            <1>         ; invalid file name chars !
2772 0000BFD3 B81A000000        <1>         mov     eax, ERR_INV_FILE_NAME  ; 26
2773 0000BFD8 EB24              <1>         jmp     short msftdf_df_stc_retn
2774                            <1>
2775                            <1> msftdf_convert_df_direntry_name:
2776                            <1>         ; mov  esi, DestinationFile_Name ; *
2777 0000BFDA BF[96950100]      <1>         mov     edi, DestinationFile_DirEntry
2778 0000BFDF E8B3F5FFFF        <1>         call    convert_file_name
2779 0000BFE4 EB1A              <1>         jmp     short msftdf_restore_current_dir_1
2780                            <1>
2781                            <1> msftdf_df_copy_sf_name:
2782 0000BFE6 89F7              <1>         mov     edi, esi
2783 0000BFE8 57                <1>         push    edi
2784 0000BFE9 BE[06950100]      <1>         mov     esi, SourceFile_Name
2785 0000BFEE B90C000000        <1>         mov     ecx, 12
2786                            <1> msftdf_df_copy_sf_name_loop:
2787 0000BFF3 AC                <1>         lodsb
2788 0000BFF4 AA                <1>         stosb
2789 0000BFF5 08C0              <1>         or      al, al
2790 0000BFF7 7402              <1>         jz      short msftdf_df_copy_sf_name_ok
2791 0000BFF9 E2F8              <1>         loop    msftdf_df_copy_sf_name_loop
2792                            <1> msftdf_df_copy_sf_name_ok:
2793 0000BFFB 5E                <1>         pop     esi
2794 0000BFFC EBB7              <1>         jmp     short msftdf_df_find_2
2795                            <1>
2796                            <1> msftdf_df_stc_retn:
2797 0000BFFE F9                <1>         stc
2798                            <1> msftdf_restore_cdir_failed:
2799                            <1> msftdf_df_error_retn:
2800 0000BFFF C3                <1>         retn
2801                            <1>
2802                            <1> msftdf_restore_current_dir_1:
2803 0000C000 803D[A6400100]00  <1>         cmp     byte [Restore_CDIR], 0
2804 0000C007 760D              <1>         jna     short msftdf_sf_check_directory
2805 0000C009 8B35[C8950100]    <1>         mov     esi, [msftdf_drv_offset]
2806 0000C00F E891C1FFFF        <1>         call    restore_current_directory
2807 0000C014 72E9              <1>         jc      short msftdf_restore_cdir_failed
2808                            <1>
2809                            <1> msftdf_sf_check_directory:
2810 0000C016 BE[C5940100]      <1>         mov     esi, SourceFile_Directory
2811 0000C01B 803E20            <1>         cmp     byte [esi], 20h
2812 0000C01E 760F              <1>         jna     short msftdf_sf_find
2813                            <1> msftdf_sf_change_directory:
2814 0000C020 FE05[A6400100]    <1>         inc     byte [Restore_CDIR]
2815 0000C026 30E4              <1>         xor     ah, ah ; CD_COMMAND sign -> 0
2816 0000C028 E81FF0FFFF        <1>         call    change_current_directory
2817 0000C02D 7227              <1>         jc      short msftdf_return
2818                            <1>
2819                            <1> ;msftdf_sf_change_prompt_dir_string:
2820                            <1> ;       call    change_prompt_dir_string
2821                            <1>
2822                            <1> msftdf_sf_find:
2823 0000C02F BE[06950100]      <1>         mov     esi, SourceFile_Name  ; Offset 66
2824 0000C034 66B80018          <1>         mov     ax, 1800h ; Only files
2825 0000C038 E854D4FFFF        <1>         call    find_first_file
2826 0000C03D 7217              <1>         jc      short msftdf_return
2827                            <1>
2828                            <1> msftdf_sf_ambgfn_check:
2829 0000C03F 6609D2            <1>         or      dx, dx ; Ambiguous filename chars used sign (DX>0)
2830 0000C042 7407              <1>         jz      short msftdf_sf_found
2831                            <1>
2832                            <1> msftdf_ambiguous_file_name_error:
2833 0000C044 B802000000        <1>         mov     eax, 2 ; File not found error
2834 0000C049 F9                <1>         stc
2835 0000C04A C3                <1>         retn
2836                            <1>
2837                            <1> msftdf_sf_found:
2838 0000C04B 80E31F            <1>         and     bl, 1Fh ; Attributes, D-V-S-H-R
2839 0000C04E 7416              <1>         jz      short msftdf_save_sf_structure
2840                            <1>
2841                            <1> msftdf_permission_denied_retn:
2842 0000C050 B805000000        <1>         mov     eax, 05h ; Access (Permission) denied !
2843 0000C055 F9                <1>         stc
2844                            <1> msftdf_rest_cdir_err_retn:
2845                            <1> msftdf_return:
2846 0000C056 C3                <1>         retn
2847                            <1>
2848                            <1> msftdf_phase_1_return:
2849 0000C057 31C0              <1>         xor     eax, eax
2850 0000C059 A2[C6950100]      <1>         mov     [move_cmd_phase], al ; 0
2851 0000C05E FEC0              <1>         inc     al ; mov al, 1
2852 0000C060 BB[ADC00000]      <1>         mov     ebx, msftdf_df2_check_directory
2853                            <1>         ;mov    edx, 0FFFFFFFFh
```

```
2854 0000C065 C3                <1>       retn
2855                            <1>
2856                            <1> msftdf_save_sf_structure:
2857 0000C066 BE[D0930100]      <1>       mov    esi, FindFile_DirEntry
2858 0000C06B BF[16950100]      <1>       mov    edi, SourceFile_DirEntry
2859 0000C070 B908000000        <1>       mov    ecx, 8
2860 0000C075 F3A5              <1>       rep    movsd
2861                            <1>
2862                            <1> msftdf_df_copy_sf_parameters:
2863 0000C077 BE0B000000        <1>       mov    esi, 11
2864 0000C07C 89F7              <1>       mov    edi, esi
2865 0000C07E 81C6[16950100]    <1>       add    esi, SourceFile_DirEntry
2866 0000C084 81C7[96950100]    <1>       add    edi, DestinationFile_DirEntry
2867                            <1>       ;mov   ecx, 21
2868 0000C08A B115              <1>       mov    cl, 21
2869 0000C08C F3A4              <1>       rep    movsb
2870                            <1>
2871                            <1> msftdf_restore_current_dir_2:
2872 0000C08E 803D[A6400100]00  <1>       cmp    byte [Restore_CDIR], 0
2873 0000C095 760D              <1>       jna    short msftdf_df2_check_move_cmd_phase
2874 0000C097 8B35[C8950100]    <1>       mov    esi, [msftdf_drv_offset]
2875 0000C09D E803C1FFFF        <1>       call   restore_current_directory
2876 0000C0A2 72B2              <1>       jc     short msftdf_rest_cdir_err_retn
2877                            <1>
2878                            <1> msftdf_df2_check_move_cmd_phase:
2879 0000C0A4 803D[C6950100]01  <1>       cmp    byte [move_cmd_phase], 1
2880 0000C0AB 74AA              <1>       je     short msftdf_phase_1_return
2881                            <1>
2882                            <1> msftdf_df2_check_directory:
2883 0000C0AD BE[45950100]      <1>       mov    esi, DestinationFile_Directory
2884 0000C0B2 803E20            <1>       cmp    byte [esi], 20h
2885 0000C0B5 760F              <1>       jna    short msftdf_make_dfde_locate_ffe_on_directory
2886                            <1> msftdf_df2_change_directory:
2887 0000C0B7 FE05[A6400100]    <1>       inc    byte [Restore_CDIR]
2888 0000C0BD 30E4              <1>       xor    ah, ah ; CD_COMMAND sign -> 0
2889 0000C0BF E888EFFFFF        <1>       call   change_current_directory
2890 0000C0C4 7290              <1>       jc     short msftdf_return
2891                            <1>
2892                            <1> ;msftdf_df2_change_prompt_dir_string:
2893                            <1> ;     call   change_prompt_dir_string
2894                            <1>
2895                            <1> msftdf_make_dfde_locate_ffe_on_directory:
2896                            <1>       ; Current directory fcluster <> Directory buffer cluster
2897                            <1>       ; Current directory will be reloaded by
2898                            <1>       ; 'locate_current_dir_file' procedure
2899                            <1>       ;
2900                            <1>       ;xor   ax, ax
2901 0000C0C6 31C0              <1>       xor    eax, eax
2902 0000C0C8 89C1              <1>       mov    ecx, eax
2903 0000C0CA 6649              <1>       dec    cx ; FFFFh
2904                            <1>              ; CX = FFFFh -> find first deleted or free entry
2905                            <1>              ; ESI would be ASCIIZ filename address if the call
2906                            <1>              ; would not be for first free or deleted dir entry
2907 0000C0CC E8CEF1FFFF        <1>       call   locate_current_dir_file
2908 0000C0D1 733F              <1>       jnc    msftdf_make_dfde_set_ff_dir_entry
2909                            <1>
2910                            <1>       ;cmp   eax, 2
2911 0000C0D3 3C02              <1>         cmp  al, 2
2912 0000C0D5 7537              <1>       jne    short msftdf_error_retn
2913                            <1>
2914                            <1> msftdf_add_new_dir_entry_check_fs:
2915 0000C0D7 8B35[C8950100]    <1>       mov    esi, [msftdf_drv_offset]
2916 0000C0DD A1[09920100]      <1>       mov    eax, [DirBuff_Cluster]
2917 0000C0E2 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
2918 0000C0E6 7711              <1>       ja     short msftdf_add_new_subdir_cluster
2919                            <1>
2920                            <1> msftdf_add_new_fs_subdir_section:
2921                            <1>       ;CL=0, CH=E5h --> deleted entry, CH=0 --> free entry
2922                            <1>         ;xor cx, cx
2923 0000C0E8 30ED              <1>       xor    ch, ch ; cx = 0 --> add a new subdir section
2924 0000C0EA E8830C0000        <1>       call   add_new_fs_section
2925 0000C0EF 721E              <1>         jc   short msftdf_dsfde_error_retn
2926                            <1>       ;mov   [createfile_LastDirCluster], eax
2927                            <1>
2928 0000C0F1 E8A30E0000        <1>       call   load_FS_sub_directory
2929                            <1>       ;mov   ebx, Directory_Buffer
2930 0000C0F6 7318              <1>       jnc    short msftdf_add_new_fs_subdir_section_ok
2931 0000C0F8 C3                <1>       retn
2932                            <1>
2933                            <1> msftdf_add_new_subdir_cluster:
2934 0000C0F9 E881150000        <1>       call   add_new_cluster
2935 0000C0FE 720F              <1>       jc     short msftdf_dsfde_error_retn
2936                            <1>
2937                            <1>       ;mov   [createfile_LastDirCluster], eax
2938                            <1>
2939 0000C100 E8570E0000        <1>       call   load_FAT_sub_directory
2940 0000C105 7309              <1>       jnc    short msftdf_add_new_subdir_cluster_ok
2941                            <1>       ; EBX = Directory buffer address
2942                            <1>
2943                            <1> msftdf_ansdc_update_parent_dir_lmdt:
2944                            <1> msftdf_make_dfde_err_upd_pdir_lmdt:
2945 0000C107 50                <1>       push   eax
2946 0000C108 E854FAFFFF        <1>       call   update_parent_dir_lmdt
2947 0000C10D 58                <1>       pop    eax
2948                            <1>
2949                            <1> msftdf_error_retn:
2950 0000C10E F9                <1>       stc
2951                            <1> msftdf_dsfde_restore_cdir_failed:
2952                            <1> msftdf_dsfde_error_retn:
2953 0000C10F C3                <1>       retn
2954                            <1>
2955                            <1> msftdf_add_new_fs_subdir_section_ok:
2956                            <1> msftdf_add_new_subdir_cluster_ok:
2957 0000C110 89DF              <1>       mov    edi, ebx ; Directory buffer address
2958                            <1>
```

```
                                <1> msftdf_make_dfde_set_ff_dir_entry:
0000C112 8B15[D88A0100]         <1>         mov    edx, [Current_Dir_FCluster]
0000C118 8915[2C960100]         <1>         mov    [createfile_FFCluster], edx
                                <1>         ; EDI = Directory entry offset
0000C11E BE[96950100]           <1>         mov    esi, DestinationFile_DirEntry
0000C123 B908000000             <1>         mov    ecx, 8
0000C128 F3A5                   <1>         rep    movsd
                                <1>
0000C12A C605[04920100]02       <1>         mov    byte [DirBuff_ValidData], 2
0000C131 E890F9FFFF             <1>         call   save_directory_buffer
0000C136 72CF                   <1>         jc     short msftdf_make_dfde_err_upd_pdir_lmdt
                                <1>
                                <1> msftdf_make_dfde_update_pdir_lmdt:
0000C138 E824FAFFFF             <1>         call   update_parent_dir_lmdt
                                <1>
                                <1> msftdf_dsfde_restore_current_dir_1:
0000C13D 803D[A6400100]00       <1>         cmp    byte [Restore_CDIR], 0
0000C144 760D                   <1>         jna    short msftdf_dsfde_check_directory
0000C146 8B35[C8950100]         <1>         mov    esi, [msftdf_drv_offset]
0000C14C E854C0FFFF             <1>         call   restore_current_directory
0000C151 72BC                   <1>         jc     short msftdf_dsfde_restore_cdir_failed
                                <1>
                                <1> msftdf_dsfde_check_directory:
0000C153 BE[C5940100]           <1>         mov    esi, SourceFile_Directory
0000C158 803E20                 <1>         cmp    byte [esi], 20h
0000C15B 760F                   <1>         jna    short msftdf_dsfde_find_file
                                <1>
                                <1> msftdf_dsfde_change_directory:
0000C15D FE05[A6400100]         <1>         inc    byte [Restore_CDIR]
0000C163 28E4                   <1>         sub    ah, ah ; CD_COMMAND sign -> 0
0000C165 E8E2EEFFFF             <1>         call   change_current_directory
0000C16A 72A3                   <1>         jc     short msftdf_dsfde_error_retn
                                <1>
                                <1> ;msftdf_dsfde_sf_change_prompt_dir_string:
                                <1> ;       call   change_prompt_dir_string
                                <1>
                                <1> msftdf_dsfde_find_file:
0000C16C BE[06950100]           <1>         mov    esi, SourceFile_Name  ; Offset 66
0000C171 668B460E               <1>         mov    ax, [esi+14] ; 80 -> SourceFile_AttributesMask
0000C175 E817D3FFFF             <1>         call   find_first_file
0000C17A 7293                   <1>         jc     short msftdf_dsfde_error_retn
                                <1>
                                <1> msftdf_dsfde_delete_direntry:
0000C17C 8B35[C8950100]         <1>         mov    esi, [msftdf_drv_offset]
                                <1>
0000C182 807E0300               <1>         cmp    byte [esi+LD_FATType], 0
0000C186 770A                   <1>         ja     short msftdf_delete_FAT_direntry
                                <1>
0000C188 30DB                   <1>         xor    bl, bl
                                <1>         ; BL = 0 -> File
                                <1>         ; EDI -> Directory buffer entry offset/address
0000C18A E8E40B0000             <1>         call   delete_fs_directory_entry
0000C18F 7315                   <1>         jnc    short msftdf_dsfde_restore_current_dir_2
0000C191 C3                     <1>         retn
                                <1>
                                <1> msftdf_delete_FAT_direntry:
0000C192 8A1D[CD930100]         <1>         mov    bl, [FindFile_LongNameEntryLength]
0000C198 668B0D[F8930100]       <1>         mov    cx, [FindFile_DirEntryNumber]
                                <1>         ; ESI = Logical DOS drive description table address
                                <1>         ; EDI = Directory buffer entry offset/address
0000C19F E89AFCFFFF             <1>         call   delete_directory_entry
0000C1A4 721C                   <1>         jc     short msftdf_retn
                                <1>
                                <1> msftdf_dsfde_restore_current_dir_2:
0000C1A6 803D[A6400100]00       <1>         cmp    byte [Restore_CDIR], 0
0000C1AD 7607                   <1>         jna    short msftdf_new_dir_fcluster_retn
                                <1>         ;mov   esi, [msftdf_drv_offset]
0000C1AF E8F1BFFFFF             <1>         call   restore_current_directory
0000C1B4 720C                   <1>         jc     short msftdf_retn
                                <1>
                                <1> msftdf_new_dir_fcluster_retn:
0000C1B6 31C9                   <1>         xor    ecx, ecx
0000C1B8 A1[2C960100]           <1>         mov    eax, [createfile_FFCluster]
0000C1BD BB[44950100]           <1>         mov    ebx, DestinationFile_Drv
                                <1>
                                <1> msftdf_retn:
0000C1C2 C3                     <1>         retn
                                <1>
                                <1>
                                <1> copy_source_file_to_destination_file:
                                <1>         ; 17/10/2016
                                <1>         ; 16/10/2016
                                <1>         ; 15/10/2016
                                <1>         ; 30/03/2016, 31/03/2016
                                <1>         ; 24/03/2016, 25/03/2016, 28/03/2016
                                <1>         ; 21/03/2016, 22/03/2016, 23/03/2016
                                <1>         ; 16/03/2016, 17/03/2016, 18/03/2016
                                <1>         ; 15/03/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>         ; 02/09/2011 (FILE.ASM 'copy_source_file_to_destination_file')
                                <1>         ; 01/08/2010 - 18/05/2011
                                <1>         ;
                                <1>         ;   Command Interpreter phase 1 enter ->
                                <1>         ;           AL = 1 -> Caller is command interpreter
                                <1>         ;           AL = 2 -> The second call, re-enter/continue
                                <1>         ;   Phase 1 -> Check source file
                                <1>         ;              'found' is required
                                <1>         ;   Phase 2 -> Check destination file,
                                <1>         ;              save 'found' or 'not found' status
                                <1>         ;              'permission denied' error will be return
                                <1>         ;              if attributes have not for ordinary file
                                <1>         ;              without readonly attribute
                                <1>         ;   Command Interpreter phase 1 return ->
                                <1>         ;              DH = Source file attributes
                                <1>         ;              DL = Destination file found status
                                <1>         ;              EAX = 0
```

```
                                <1>       ;   Command Interpreter phase 2 enter ->
                                <1>       ;             AL = 2 -> Continue from the last position
                                <1>       ;             AH =
                                <1>       ;   Phase 3 -> Load source file or use read/write cluster method
                                <1>       ;   Phase 4 -> Create destination file if it is not found
                                <1>       ;   Phase 5 -> Open destination file
                                <1>       ;   Phase 6 -> Read from source and write to destination
                                <1>       ;   Phase 7 -> Unload source file, if it is loaded at memory
                                <1>       ;       cf = 1 causes to return before the phase 7
                                <1>       ;             but loaded file will be unloaded
                                <1>       ;            (allocated memory block will be deallocated)
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;      ESI = Source File Pathname (Asciiz)
                                <1>       ;       EDI = Destination File Pathname (Asciiz)
                                <1>       ;       AL = 0 --> Interrupt (System call)
                                <1>       ;       AL > 0 --> Command Interpreter (Question)
                                <1>       ;       AL = 1 --> Question Phase
                                <1>       ;       AL = 2 --> Progress Phase
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 0 -> OK
                                <1>       ;     EAX = Destination file first cluster
                                <1>       ;
                                <1>       ;       CL > 0 if there is file reading error before EOF
                                <1>       ;             (incomplete copy)
                                <1>       ;       CH > 0 if file is (full) loaded at memory
                                <1>       ;
                                <1>       ;     cf = 1 -> Error code in AL (EAX)
                                <1>       ;
                                <1>       ; (EBX, ECX, ESI, EDI register contents will be changed)
                                <1>
                                <1>
0000C1C3 3C02                   <1>       cmp   al, 2
0000C1C5 0F845A020000           <1>       je    csftdf2_check_cdrv
                                <1>
                                <1> ; Phase 1
                                <1>
0000C1CB A2[EC950100]           <1>       mov   byte [copy_cmd_phase], al
                                <1>
0000C1D0 57                     <1>       push  edi ; *
                                <1>
                                <1> csftdf_parse_sf_path:
0000C1D1 BF[C4940100]           <1>       mov   edi, SourceFile_Drv
0000C1D6 E887F4FFFF             <1>       call  parse_path_name
0000C1DB 721C                   <1>       jc    short csftdf_parse_sf_path_failed
                                <1>
                                <1> csftdf_parse_df_path:
0000C1DD 5E                     <1>       pop   esi ; * (pushed edi)
                                <1>
                                <1> csftdf_sf_check_filename_exists:
0000C1DE 803D[06950100]21       <1>       cmp   byte [SourceFile_Name], 21h
0000C1E5 7215                   <1>       jb    short csftdf_sf_file_not_found_error
                                <1>
0000C1E7 BF[44950100]           <1>       mov   edi, DestinationFile_Drv
0000C1EC E871F4FFFF             <1>       call  parse_path_name
0000C1F1 7310                   <1>       jnc   short csftdf_check_sf_cdrv
                                <1>
0000C1F3 3C01                   <1>       cmp   al, 1 ; File or directory name is not existing
0000C1F5 760C                   <1>       jna   short csftdf_check_sf_cdrv
                                <1>
                                <1> csftdf_parse_df_path_failed:
0000C1F7 F9                     <1>       stc
                                <1> csftdf_sf_error_retn:
0000C1F8 C3                     <1>       retn
                                <1>
                                <1> csftdf_parse_sf_path_failed:
0000C1F9 5F                     <1>       pop   edi ; *
0000C1FA EBFC                   <1>       jmp   short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_file_not_found_error:
0000C1FC B802000000             <1>       mov   eax, 2 ; File not found
0000C201 EBF5                   <1>       jmp   short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_check_sf_cdrv:
0000C203 8A3D[DE8A0100]         <1>       mov   bh, [Current_Drv]
                                <1>
0000C209 883D[EF950100]         <1>       mov   [csftdf_cdrv], bh ; 23/03/2016
                                <1>
0000C20F 8A15[C4940100]         <1>       mov   dl, [SourceFile_Drv]
0000C215 38FA                   <1>       cmp   dl, bh ; byte [Current_Drv]
0000C217 7407                   <1>       je    short csftdf_sf_check_directory
                                <1>
0000C219 E8D0BEFFFF             <1>       call  change_current_drive
0000C21E 72D8                   <1>       jc    short csftdf_sf_error_retn
                                <1>
                                <1> csftdf_sf_check_directory:
0000C220 BE[C5940100]           <1>       mov   esi, SourceFile_Directory
0000C225 803E20                 <1>       cmp   byte [esi], 20h
0000C228 760F                   <1>       jna   short csftdf_find_sf
                                <1>
                                <1> csftdf_sf_change_directory:
0000C22A FE05[A6400100]         <1>       inc   byte [Restore_CDIR]
0000C230 30E4                   <1>       xor   ah, ah ; CD_COMMAND sign -> 0
0000C232 E815EEFFFF             <1>       call  change_current_directory
0000C237 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:
0000C239 BE[06950100]           <1>       mov   esi, SourceFile_Name
0000C23E 66B80018               <1>       mov   ax, 1800h ; Except volume label and dirs
0000C242 E84AD2FFFF             <1>       call  find_first_file
0000C247 72AF                   <1>       jc    short csftdf_sf_error_retn
```

```
3169                                    <1>
3170                                    <1> csftdf_sf_ambgfn_check:
3171 0000C249 6621D2                   <1>         and     dx, dx ; Ambiguous filename chars used sign (DX>0)
3172 0000C24C 7407                     <1>         jz      short csftdf_sf_found
3173                                    <1>
3174                                    <1> csftdf_ambiguous_file_name_error:
3175 0000C24E B802000000               <1>         mov     eax, 2 ; File not found error
3176 0000C253 F9                       <1>         stc
3177 0000C254 C3                       <1>         retn
3178                                    <1>
3179                                    <1> csftdf_sf_found:
3180 0000C255 A3[F0950100]             <1>         mov     [csftdf_filesize], eax
3181                                    <1>
3182 0000C25A 09C0                     <1>         or      eax, eax
3183 0000C25C 7507                     <1>         jnz     short csftdf_set_source_file_direntry
3184                                    <1>
3185                                    <1> csftdf_sf_file_size_zero:
3186 0000C25E B814000000               <1>         mov     eax, 20 ; TRDOS zero length (file size) error
3187 0000C263 F9                       <1>         stc
3188 0000C264 C3                       <1>         retn
3189                                    <1>
3190                                    <1> csftdf_set_source_file_direntry:
3191 0000C265 BE[D0930100]             <1>         mov     esi, FindFile_DirEntry
3192 0000C26A BF[16950100]             <1>         mov     edi, SourceFile_DirEntry
3193 0000C26F B908000000               <1>         mov     ecx, 8
3194 0000C274 F3A5                     <1>         rep     movsd
3195                                    <1>
3196                                    <1> csftdf_sf_restore_cdrv:
3197                                    <1>         ; 22/03/2016
3198 0000C276 8A15[EF950100]           <1>         mov     dl, [csftdf_cdrv]
3199 0000C27C 3A15[DE8A0100]           <1>         cmp     dl, [Current_Drv]
3200 0000C282 7407                     <1>         je      short csftdf_sf_restore_cdir
3201 0000C284 E865BEFFFF               <1>         call    change_current_drive
3202 0000C289 724F                     <1>         jc      short csftdf_df_error_retn ; 30/03/2016
3203                                    <1>
3204                                    <1> csftdf_sf_restore_cdir:
3205 0000C28B 803D[A6400100]00         <1>         cmp     byte [Restore_CDIR], 0
3206 0000C292 7612                     <1>         jna     short csftdf_df_check_filename_exists
3207 0000C294 29C0                     <1>         sub     eax, eax
3208 0000C296 BE00010900               <1>         mov     esi, Logical_DOSDisks
3209 0000C29B 88D4                     <1>         mov     ah, dl ; byte [csftdf_cdrv]
3210 0000C29D 01C6                     <1>         add     esi, eax
3211 0000C29F E801BFFFFF               <1>         call    restore_current_directory
3212 0000C2A4 7234                     <1>         jc      short csftdf_df_error_retn
3213                                    <1>
3214                                    <1> csftdf_df_check_filename_exists:
3215 0000C2A6 803D[86950100]20         <1>         cmp     byte [DestinationFile_Name], 20h
3216 0000C2AD 7716                     <1>         ja      short csftdf_check_df_cdrv
3217                                    <1>
3218                                    <1> csftdf_copy_sf_name:
3219 0000C2AF BF[86950100]             <1>         mov     edi, DestinationFile_Name
3220 0000C2B4 BE[06950100]             <1>         mov     esi, SourceFile_Name
3221 0000C2B9 B10C                     <1>         mov     cl, 12
3222                                    <1>
3223                                    <1> csftdf_df_copy_sf_name_loop:
3224 0000C2BB AC                       <1>         lodsb
3225 0000C2BC AA                       <1>         stosb
3226 0000C2BD 08C0                     <1>         or      al, al
3227 0000C2BF 7404                     <1>         jz      short csftdf_check_df_cdrv
3228 0000C2C1 FEC9                     <1>         dec     cl
3229 0000C2C3 75F6                     <1>         jnz     csftdf_df_copy_sf_name_loop
3230                                    <1>
3231                                    <1> csftdf_check_df_cdrv:
3232 0000C2C5 8A15[44950100]           <1>         mov     dl, [DestinationFile_Drv]
3233 0000C2CB 3A15[DE8A0100]           <1>         cmp     dl, [Current_Drv]
3234 0000C2D1 7408                     <1>         je      short csftdf_df_check_directory
3235                                    <1>
3236 0000C2D3 E816BEFFFF               <1>         call    change_current_drive
3237 0000C2D8 7301                     <1>         jnc     short csftdf_df_check_directory
3238                                    <1>
3239                                    <1> csftdf_df_error_retn:
3240 0000C2DA C3                       <1>         retn
3241                                    <1>
3242                                    <1> csftdf_df_check_directory:
3243 0000C2DB BE[45950100]             <1>         mov     esi, DestinationFile_Directory
3244 0000C2E0 803E20                   <1>           cmp     byte [esi], 20h
3245 0000C2E3 760F                     <1>         jna     short csftdf_find_df
3246                                    <1>
3247                                    <1> csftdf_df_change_directory:
3248 0000C2E5 FE05[A6400100]           <1>         inc     byte [Restore_CDIR]
3249 0000C2EB 28E4                     <1>         sub     ah, ah ; CD_COMMAND sign -> 0
3250 0000C2ED E85AEDFFFF               <1>         call    change_current_directory
3251 0000C2F2 72E6                     <1>         jc      short csftdf_df_error_retn
3252                                    <1>
3253                                    <1> ;csftdf_df_change_prompt_dir_string:
3254                                    <1> ;       call    change_prompt_dir_string
3255                                    <1>
3256                                    <1> csftdf_find_df:
3257                                    <1>         ; 23/03/2016
3258 0000C2F4 29DB                     <1>         sub     ebx, ebx
3259 0000C2F6 8A3D[44950100]           <1>         mov     bh,  [DestinationFile_Drv]
3260 0000C2FC 81C300010900             <1>         add     ebx, Logical_DOSDisks
3261 0000C302 891D[1C960100]           <1>         mov     [csftdf_df_drv_dt], ebx
3262                                    <1>
3263 0000C308 BE[86950100]             <1>         mov     esi, DestinationFile_Name
3264 0000C30D 6631C0                   <1>         xor     ax, ax
3265                                    <1>                 ; DestinationFile_AttributesMask -> any/zero
3266 0000C310 E87CD1FFFF               <1>         call    find_first_file
3267 0000C315 7218                     <1>         jc      short csftdf_df_check_error_code
3268                                    <1>
3269                                    <1> csftdf_df_ambgfn_check:
3270 0000C317 6609D2                   <1>         or      dx, dx ; Ambiguous filename chars used sign (DX>0)
3271 0000C31A 752A                     <1>         jnz     short csftdf_df_error_inv_fname
3272                                    <1>
3273                                    <1> csftdf_df_found:
```

```
3274 0000C31C C605[EE950100]01    <1>       mov    byte [DestinationFileFound], 1
3275                              <1>       ; 17/10/2016 (cl -> bl)
3276 0000C323 80E31F              <1>       and    bl, 1Fh ; Attributes, D-V-S-H-R
3277 0000C326 745F                <1>       jz     short csftdf_df_save_first_cluster
3278                              <1>
3279                              <1> csftdf_df_permission_denied_retn:
3280 0000C328 B805000000          <1>       mov    eax, 05h ; Access/Permission denied.
3281                              <1> csftdf_df_error_stc_retn:
3282 0000C32D F9                  <1>       stc
3283 0000C32E C3                  <1>       retn
3284                              <1>
3285                              <1> csftdf_df_check_error_code:
3286                              <1>       ;cmp   eax, 2
3287 0000C32F 3C02                <1>       cmp    al, 2
3288 0000C331 75FA                <1>       jne    short csftdf_df_error_stc_retn
3289                              <1>
3290 0000C333 C605[EE950100]00    <1>       mov    byte [DestinationFileFound], 0
3291                              <1>
3292                              <1>       ; 15/10/2016
3293 0000C33A BE[C0930100]        <1>       mov    esi, FindFile_Name ; *
3294 0000C33F E80BD5FFFF          <1>       call   check_filename
3295 0000C344 7307                <1>       jnc    short csftdf_df_valid_fname
3296                              <1> csftdf_df_error_inv_fname: ; 'invalid file name !'
3297 0000C346 B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME  ; 26
3298 0000C34B F9                  <1>       stc
3299 0000C34C C3                  <1>       retn
3300                              <1>
3301                              <1> csftdf_df_valid_fname:
3302                              <1>       ; 21/03/2016
3303                              <1>       ; (Capitalized file name)
3304                              <1>       ;mov   esi, FindFile_Name ; * ; 15/10/2016
3305 0000C34D BF[86950100]        <1>       mov    edi, DestinationFile_Name
3306 0000C352 A5                  <1>       movsd
3307 0000C353 A5                  <1>       movsd
3308 0000C354 A5                  <1>       movsd
3309                              <1>       ;movsb
3310                              <1>
3311                              <1> csftdf_check_disk_free_size_0:
3312 0000C355 A1[32950100]        <1>       mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
3313                              <1>
3314                              <1> csftdf_check_disk_free_size_1:
3315                              <1>       ;sub   ebx, ebx
3316                              <1>       ;mov   esi, Logical_DOSDisks
3317                              <1>       ;mov   bh,  [DestinationFile_Drv]
3318                              <1>       ;add   esi, ebx
3319                              <1>
3320 0000C35A 8B35[1C960100]      <1>       mov    esi, [csftdf_df_drv_dt] ; 23/03/2016
3321                              <1>
3322 0000C360 0FB74E11            <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
3323 0000C364 01C8                <1>       add    eax, ecx
3324 0000C366 48                  <1>       dec    eax  ; file size (additional bytes) + 511 (round up)
3325                              <1> csftdf_check_disk_free_size_3: ; 16/03/2016
3326 0000C367 29D2                <1>       sub    edx, edx
3327 0000C369 F7F1                <1>       div    ecx ; bytes per sector
3328                              <1>
3329                              <1> csftdf_check_disk_free_size:
3330 0000C36B 3B4674              <1>       cmp    eax, [esi+LD_FreeSectors]
3331 0000C36E 0F8294000000        <1>         jb     csftdf_check_disk_free_size_ok
3332 0000C374 770A                <1>       ja     short csftdf_df_insufficient_disk_space
3333                              <1>
3334 0000C376 807E0300            <1>       cmp    byte [esi+LD_FATType], 0 ; FS needs FDT sector also.
3335 0000C37A 0F8788000000        <1>         ja     csftdf_check_disk_free_size_ok
3336                              <1>
3337                              <1> csftdf_df_insufficient_disk_space:
3338 0000C380 B827000000          <1>       mov    eax, 27h ; insufficient disk space
3339 0000C385 EBA6                <1>       jmp    short csftdf_df_error_stc_retn
3340                              <1>
3341                              <1> csftdf_df_save_first_cluster:
3342                              <1>       ; ESI = FindFile_DirEntry (for the old destination file)
3343                              <1>       ; EAX = Old destination file size
3344                              <1>       ; 24/03/2016
3345                              <1>       ; EDI = Directory entry address (within Dir Buffer boundaries)
3346 0000C387 81EF00000800        <1>       sub    edi, Directory_Buffer  ; (<65536)
3347 0000C38D 66C1EF05            <1>       shr    di, 5 ; Convert entry offset to entry index/number
3348 0000C391 66893D[BE950100]    <1>       mov    [DestinationFile_DirEntryNumber], di ; (<2048)
3349                              <1>
3350                              <1> csftdf_df_check_sf_df_fcluster:
3351 0000C398 668B5614            <1>       mov    dx, [esi+DirEntry_FstClusHI]
3352 0000C39C C1E210              <1>       shl    edx, 16
3353 0000C39F 668B561A            <1>       mov    dx, [esi+DirEntry_FstClusLO]
3354 0000C3A3 8915[00960100]      <1>       mov    [csftdf_df_cluster], edx
3355                              <1> csftdf_df_check_sf_df_fcluster_1:
3356 0000C3A9 668B15[2A950100]    <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
3357 0000C3B0 C1E210              <1>       shl    edx, 16
3358 0000C3B3 668B15[30950100]    <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
3359 0000C3BA 3B15[00960100]      <1>       cmp    edx, [csftdf_df_cluster]
3360 0000C3C0 7512                <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
3361                              <1> csftdf_df_check_sf_df_drv:
3362 0000C3C2 8A15[C4940100]      <1>       mov    dl, [SourceFile_Drv]
3363 0000C3C8 3A15[44950100]      <1>       cmp    dl, [DestinationFile_Drv]
3364 0000C3CE 7504                <1>       jne    short csftdf_df_check_sf_df_fcluster_ok
3365                              <1>
3366                              <1>       ; source and destination files are same !
3367                              <1>       ; (they have same first cluster value on same logical disk)
3368                              <1>
3369 0000C3D0 31C0                <1>       xor    eax, eax ; mov eax, 0 -> Bad command or file name !
3370 0000C3D2 F9                  <1>       stc
3371 0000C3D3 C3                  <1>       retn
3372                              <1>
3373                              <1> csftdf_df_check_sf_df_fcluster_ok:
3374                              <1> csftdf_df_move_findfile_struct:
3375                              <1>       ; mov  esi, FindFile_DirEntry
3376 0000C3D4 BF[96950100]        <1>       mov    edi, DestinationFile_DirEntry
3377 0000C3D9 B908000000          <1>       mov    ecx, 8
3378 0000C3DE F3A5                <1>       rep    movsd
```

```
                                <1>
                                <1> csftdf_check_disk_free_size_2:
0000C3E0 89C2                   <1>         mov     edx, eax ; Old destination file size
                                <1>
                                <1>         ;mov    eax, [SourceFile_DirEntry+DirEntry_FileSize]
0000C3E2 A1[F0950100]           <1>         mov     eax, [csftdf_filesize] ; 23/03/2016
                                <1>
                                <1>         ;;sub   ecx, ecx ; 0
                                <1>         ;mov    esi, Logical_DOSDisks
                                <1>         ;mov    ch,  [DestinationFile_Drv]
                                <1>         ;add    esi, ecx
                                <1>         ;
                                <1>         ;mov    [csftdf_df_drv_dt], esi
                                <1>
0000C3E7 8B35[1C960100]         <1>         mov     esi, [csftdf_df_drv_dt] ; 23/03/2016
                                <1>
0000C3ED 668B4E11               <1>         mov     cx, [esi+LD_BPB+BytesPerSec] ; 17, LD_BPB + 0Bh
0000C3F1 01CA                   <1>         add     edx, ecx ; + 512
0000C3F3 01C8                   <1>         add     eax, ecx ; + 512
0000C3F5 4A                     <1>         dec     edx ; old file size + 511 (round up)
0000C3F6 48                     <1>         dec     eax ; new file size + 511 (round up)
0000C3F7 F7D9                   <1>         neg     ecx ; -512 ; 0FFFFFE00h
0000C3F9 21CA                   <1>         and     edx, ecx ; = old sector count * 512
0000C3FB 21C8                   <1>         and     eax, ecx ; = new sector count * 512
                                <1>
0000C3FD 29D0                   <1>         sub     eax, edx ; new file size - old file size (on disk)
0000C3FF 7607                   <1>         jna     short csftdf_check_disk_free_size_ok
                                <1>
0000C401 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)
0000C403 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:
0000C408 A0[EC950100]           <1>         mov     al, [copy_cmd_phase]
0000C40D 3C01                   <1>         cmp     al, 1
0000C40F 7514                   <1>         jne     short csftdf2_check_cdrv
                                <1>
0000C411 31C0                   <1>         xor     eax, eax
0000C413 A2[EC950100]           <1>         mov     [copy_cmd_phase], al ; 0
                                <1>
0000C418 8A15[EE950100]         <1>         mov     dl, [DestinationFileFound]
0000C41E 8A35[21950100]         <1>         mov     dh, [SourceFile_DirEntry+11] ; Attributes
                                <1>
                                <1> csftdf_return:
0000C424 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
0000C425 803D[EE950100]00       <1>         cmp     byte [DestinationFileFound], 0
0000C42C 7739                   <1>         ja      short csftdf2_set_sf_percentage
                                <1>
                                <1> csftdf2_create_file:
0000C42E BE[86950100]           <1>         mov     esi, DestinationFile_Name
0000C433 A1[F0950100]           <1>         mov     eax, [csftdf_filesize]
0000C438 30C9                   <1>         xor     cl, cl ; 0
                                <1>
0000C43A 31DB                   <1>         xor     ebx, ebx ; 0
0000C43C 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)
```

```
3484                                    <1>       ;
3485                                    <1>       ;      cf = 1 -> error code in AL (EAX)
3486                                    <1>
3487 0000C43D E8EC050000                <1>       call   create_file
3488                                    <1>       ;pop   esi
3489 0000C442 0F82A3050000              <1>         jc     csftdf2_rw_error
3490                                    <1>
3491                                    <1> csftdf2_create_file_OK:
3492 0000C448 A3[00960100]              <1>       mov    [csftdf_df_cluster], eax
3493                                    <1>
3494                                    <1>       ; 24/03/2016
3495 0000C44D 668915[BE950100]          <1>       mov    [DestinationFile_DirEntryNumber], dx
3496                                    <1>
3497                                    <1>       ; 21/03/2016
3498 0000C454 BE00000800                <1>       mov    esi, Directory_Buffer
3499 0000C459 C1E205                    <1>       shl    edx, 5 ; 32 * index number
3500 0000C45C 01D6                      <1>       add    esi, edx
3501 0000C45E BF[96950100]              <1>       mov    edi, DestinationFile_DirEntry
3502 0000C463 B108                      <1>       mov    cl, 8 ; 32 bytes
3503 0000C465 F3A5                      <1>       rep    movsd
3504                                    <1>
3505                                    <1> csftdf2_set_sf_percentage:
3506                                    <1>       ; 17/03/2016
3507 0000C467 31C0                      <1>       xor    eax, eax
3508 0000C469 A2[14960100]              <1>       mov    [csftdf_percentage], al ; 0, reset
3509                                    <1>
3510 0000C46E A3[0C960100]              <1>       mov    [csftdf_sf_rbytes], eax ; 0, reset
3511 0000C473 A3[10960100]              <1>       mov    [csftdf_df_wbytes], eax ; 0, reset
3512                                    <1>
3513 0000C478 8A25[C4940100]            <1>       mov    ah, [SourceFile_Drv]
3514 0000C47E BE00010900                <1>       mov    esi, Logical_DOSDisks
3515 0000C483 01C6                      <1>       add    esi, eax
3516                                    <1>
3517 0000C485 8935[18960100]            <1>       mov    [csftdf_sf_drv_dt], esi ; 23/03/2016
3518                                    <1>
3519 0000C48B 668B15[2A950100]          <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusHI]
3520 0000C492 C1E210                    <1>       shl    edx, 16
3521 0000C495 668B15[30950100]          <1>       mov    dx, [SourceFile_DirEntry+DirEntry_FstClusLO]
3522 0000C49C 8915[FC950100]            <1>       mov    [csftdf_sf_cluster], edx
3523                                    <1>
3524                                    <1>       ; 16/03/2016
3525                                    <1>       ; Note: Singlix FS boot sector parameters (for cluster
3526                                    <1>       ;      related calculations) has same offset
3527                                    <1>       ;      values from LD_BPB as in FAT file system.
3528                                    <1>       ;      [esi+LD_BPB+SecPerClust] is 1 for Singlix FS.
3529                                    <1>       ;
3530 0000C4A2 0FB64E13                  <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
3531 0000C4A6 880D[42950100]            <1>       mov    [SourceFile_SecPerClust], cl
3532                                    <1>
3533                                    <1>       ; 17/03/2016
3534 0000C4AC 386E03                    <1>       cmp    [esi+LD_FATType], ch ; 0
3535 0000C4AF 7707                      <1>       ja     short csftdf2_set_sf_percent_rsize1
3536                                    <1>
3537 0000C4B1 B800000100                <1>       mov    eax, 65536 ; read/write buffer size for Singlix FS
3538 0000C4B6 EB06                      <1>       jmp    short csftdf2_set_sf_percent_rsize2
3539                                    <1>
3540                                    <1> csftdf2_set_sf_percent_rsize1:
3541 0000C4B8 668B4611                  <1>       mov    ax, [esi+LD_BPB+BytesPerSec]
3542 0000C4BC F7E1                      <1>       mul    ecx
3543                                    <1>       ;sub   edx, edx
3544                                    <1> csftdf2_set_sf_percent_rsize2:
3545 0000C4BE A3[04960100]              <1>       mov    [csftdf_r_size], eax
3546                                    <1>
3547                                    <1> csftdf2_set_df_percentage:
3548                                    <1>       ;sub   eax, eax
3549                                    <1>       ;mov   ah, [DestinationFile_Drv]
3550                                    <1>       ;mov   edi, Logical_DOSDisks
3551                                    <1>       ;add   edi, eax
3552                                    <1>       ;mov   [csftdf_df_drv_dt], edi ; 17/03/2016
3553                                    <1>
3554 0000C4C3 8B3D[1C960100]            <1>       mov    edi, [csftdf_df_drv_dt] ; 23/03/2016
3555                                    <1>
3556                                    <1>       ; 16/03/2016
3557                                    <1>       ; Note: Singlix FS boot sector parameters (for cluster
3558                                    <1>       ;      related calculations) has same offset
3559                                    <1>       ;      values from LD_BPB as in FAT file system.
3560                                    <1>       ;      [edi+LD_BPB+SecPerClust] is 1 for Singlix FS.
3561                                    <1>       ;
3562                                    <1>       ;movzx ecx, byte [edi+LD_BPB+SecPerClust]
3563 0000C4C9 8A4F13                    <1>       mov    cl, [edi+LD_BPB+SecPerClust]
3564 0000C4CC 880D[C2950100]            <1>       mov    [DestinationFile_SecPerClust], cl
3565                                    <1>
3566                                    <1>       ; 17/03/2016
3567 0000C4D2 386F03                    <1>       cmp    [edi+LD_FATType], ch ; 0
3568 0000C4D5 7707                      <1>       ja     short csftdf2_set_df_percent_wsize1
3569                                    <1>
3570 0000C4D7 B800000100                <1>       mov    eax, 65536 ; read/write buffer size for Singlix FS
3571 0000C4DC EB06                      <1>       jmp    short csftdf2_set_df_percent_wsize2
3572                                    <1>
3573                                    <1> csftdf2_set_df_percent_wsize1:
3574 0000C4DE 0FB74711                  <1>       movzx  eax, word [edi+LD_BPB+BytesPerSec]
3575 0000C4E2 F7E1                      <1>       mul    ecx
3576                                    <1>       ;sub   edx, edx
3577                                    <1> csftdf2_set_df_percent_wsize2:
3578 0000C4E4 A3[08960100]              <1>       mov    [csftdf_w_size], eax
3579                                    <1>
3580 0000C4E9 A1[F0950100]              <1>       mov    eax, [csftdf_filesize]
3581                                    <1>
3582 0000C4EE 3D00000100                <1>       cmp    eax, 65536 ; 64KB   ; small file
3583 0000C4F3 721F                      <1>       jb     short csftdf2_load_file ; do not display percentage
3584                                    <1>
3585                                    <1> csftdf2_reset_wf_percent_ptr_chk_64k:
3586 0000C4F5 B201                      <1>       mov    dl, 1 ; 25/03/2016
3587                                    <1>
3588 0000C4F7 3D00000400                <1>       cmp    eax, 65536*4 ; 256KB
```

```
3589 0000C4FC 7310                <1>       jnb    short csftdf2_enable_percentage_display ; big file
3590                              <1>
3591                              <1>       ; 64-128KB file size for floppy disks
3592 0000C4FE 3815[C4940100]      <1>       cmp    byte [SourceFile_Drv], dl ; 1 ; read from floppy disk ?
3593 0000C504 7608                <1>       jna    short csftdf2_enable_percentage_display
3594                              <1>
3595 0000C506 3815[44950100]      <1>       cmp    byte [DestinationFile_Drv], dl ; 1 ; write to floppy disk ?
3596 0000C50C 7706                <1>       ja     short csftdf2_load_file
3597                              <1>
3598                              <1> csftdf2_enable_percentage_display:
3599 0000C50E 8815[14960100]      <1>       mov    [csftdf_percentage], dl ; 1
3600                              <1>
3601                              <1> csftdf2_load_file:
3602                              <1>       ; 13/05/2016
3603                              <1>       ; 19/03/2016
3604                              <1>       ; 18/03/2016
3605                              <1>       ; 17/03/2016
3606 0000C514 B40F                <1>       mov    ah, 0Fh
3607 0000C516 E85652FFFF          <1>       call   _int10h
3608                              <1>       ; 13/05/2016
3609 0000C51B 883D[15960100]      <1>       mov    [csftdf_videopage], bh ; active video page
3610 0000C521 B403                <1>       mov    ah, 03h
3611 0000C523 E84952FFFF          <1>       call   _int10h
3612 0000C528 668915[16960100]    <1>       mov    [csftdf_cursorpos], dx
3613                              <1>
3614 0000C52F 29C0                <1>       sub    eax, eax
3615 0000C531 A2[ED950100]        <1>       mov    [csftdf_rw_err], al ; 0
3616                              <1>
3617                              <1> ; ///
3618                              <1> csftdf_sf_amb: ; 15/03/2016
3619 0000C536 8B0D[F0950100]      <1>       mov    ecx, [csftdf_filesize]    ; 23/03/2016
3620                              <1>
3621                              <1>       ; TRDOS 386 (TRDOS v2.0)
3622                              <1>       ; Allocate contiguous memory block for loading the file
3623                              <1>
3624                              <1>       ;mov   ecx, [SourceFile_DirEntry+DirEntry_FileSize]
3625                              <1>
3626                              <1>       ;sub   eax, eax ; First free memory aperture
3627                              <1>
3628                              <1>       ; eax = 0 (Allocate memory from the beginning)
3629                              <1>       ; ecx = File (Allocation) size in bytes
3630                              <1>
3631 0000C53C E8449FFFFF          <1>       call   allocate_memory_block
3632 0000C541 7304                <1>       jnc    short loc_check_sf_save_loading_parms
3633                              <1>
3634 0000C543 29C0                <1>       sub    eax, eax
3635 0000C545 29C9                <1>       sub    ecx, ecx
3636                              <1>
3637                              <1> loc_check_sf_save_loading_parms:
3638 0000C547 A3[F4950100]        <1>       mov    [csftdf_sf_mem_addr], eax ; loading address
3639 0000C54C 890D[F8950100]      <1>       mov    [csftdf_sf_mem_bsize], ecx ; block size
3640                              <1> ; ///
3641                              <1>       ; 19/03/2016
3642 0000C552 8B35[18960100]      <1>       mov    esi, [csftdf_sf_drv_dt] ; logical dos drv desc. tbl.
3643                              <1>
3644                              <1>       ; 17/03/2016
3645 0000C558 09C0                <1>       or     eax, eax ; contiguous free memory block address
3646 0000C55A 0F845B010000        <1>         jz     csftdf2_read_sf_cluster
3647                              <1>
3648                              <1>       ; 18/03/2016
3649 0000C560 8B1D[F4950100]      <1>       mov    ebx, [csftdf_sf_mem_addr] ; memory block address
3650                              <1>
3651 0000C566 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
3652 0000C56A 0F8605020000        <1>         jna    csftdf2_load_fs_file
3653                              <1>
3654                              <1> csftdf2_load_fat_file:
3655 0000C570 53                  <1>       push   ebx ; *
3656                              <1>
3657                              <1> csftdf2_load_fat_file_next:
3658 0000C571 BE[F5460100]        <1>       mov    esi, msg_reading
3659 0000C576 E8EAAFFFFF          <1>       call   print_msg
3660                              <1>
3661 0000C57B 803D[14960100]00    <1>       cmp    byte [csftdf_percentage], 0
3662 0000C582 7605                <1>       jna    short csftdf2_load_fat_file_1
3663                              <1>
3664 0000C584 E87C000000          <1>       call   csftdf2_print_percentage ; 19/03/2016
3665                              <1>
3666                              <1> csftdf2_load_fat_file_1:
3667 0000C589 8B35[18960100]      <1>       mov    esi, [csftdf_sf_drv_dt]
3668 0000C58F 5B                  <1>       pop    ebx ; *
3669                              <1>
3670                              <1> csftdf2_load_fat_file_2:
3671 0000C590 E8B8000000          <1>       call   csftdf2_read_fat_file_sectors ; 19/03/2016
3672 0000C595 0F8250040000        <1>         jc     csftdf2_rw_error ; eocc! or disk error!
3673                              <1>
3674 0000C59B 09D2                <1>       or     edx, edx ; edx > 0 -> EOF
3675 0000C59D 7520                <1>       jnz    short csftdf2_load_fat_file_ok
3676                              <1>
3677 0000C59F 803D[14960100]00    <1>       cmp    byte [csftdf_percentage], 0
3678 0000C5A6 76E8                <1>       jna    short csftdf2_load_fat_file_2
3679                              <1>
3680 0000C5A8 53                  <1>       push   ebx ; *
3681                              <1>
3682                              <1>       ; Set cursor position
3683                              <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
3684 0000C5A9 8A3D[15960100]      <1>       mov    bh, [csftdf_videopage]
3685 0000C5AF 668B15[16960100]    <1>       mov    dx, [csftdf_cursorpos]
3686 0000C5B6 B402                <1>       mov    ah, 2
3687 0000C5B8 E8B451FFFF          <1>       call   _int10h
3688 0000C5BD EBB2                <1>       jmp    short csftdf2_load_fat_file_next
3689                              <1>
3690                              <1> csftdf2_load_fat_file_ok:
3691 0000C5BF 803D[14960100]00    <1>       cmp    byte [csftdf_percentage], 0
3692 0000C5C6 0F8651020000        <1>         jna    csftdf2_save_file ; 25/03/2016
3693                              <1>
```

```
3694                                    <1>       ; "Reading... 100%"
3695 0000C5CC BF[0D470100]             <1>       mov    edi, percentagestr
3696 0000C5D1 B031                     <1>       mov    al, '1'
3697 0000C5D3 AA                       <1>       stosb
3698 0000C5D4 B030                     <1>       mov    al, '0'
3699 0000C5D6 AA                       <1>       stosb
3700 0000C5D7 AA                       <1>       stosb
3701                                    <1>
3702 0000C5D8 8A3D[15960100]           <1>       mov    bh, [csftdf_videopage]
3703 0000C5DE 668B15[16960100]         <1>       mov    dx, [csftdf_cursorpos]
3704 0000C5E5 B402                     <1>       mov    ah, 2
3705 0000C5E7 E88551FFFF               <1>       call   _int10h
3706                                    <1>
3707 0000C5EC BE[F5460100]             <1>       mov    esi, msg_reading
3708 0000C5F1 E86FAFFFFF               <1>       call   print_msg
3709                                    <1>
3710 0000C5F6 BE[0D470100]             <1>       mov    esi, percentagestr
3711 0000C5FB E865AFFFFF               <1>       call   print_msg
3712                                    <1>
3713 0000C600 E918020000               <1>         jmp     csftdf2_save_file ; 25/03/2016
3714                                    <1>
3715                                    <1> csftdf2_print_percentage:
3716                                    <1>       ; 09/12/2017
3717                                    <1>       ; 19/03/2016
3718                                    <1>       ; 18/03/2016
3719 0000C605 B020                     <1>       mov    al, 20h
3720 0000C607 BF[0D470100]             <1>       mov    edi, percentagestr
3721 0000C60C AA                       <1>       stosb
3722 0000C60D AA                       <1>       stosb
3723 0000C60E A1[0C960100]             <1>       mov    eax, [csftdf_sf_rbytes]
3724 0000C613 BA64000000               <1>       mov    edx, 100
3725 0000C618 F7E2                     <1>       mul    edx
3726 0000C61A 8B0D[F0950100]           <1>       mov    ecx, [csftdf_filesize]
3727 0000C620 F7F1                     <1>       div    ecx
3728 0000C622 B10A                     <1>       mov    cl, 10
3729 0000C624 F6F1                     <1>       div    cl
3730 0000C626 80C430                   <1>       add    ah, '0'
3731 0000C629 8827                     <1>       mov    [edi], ah
3732 0000C62B 20C0                     <1>       and    al, al
3733 0000C62D 740A                     <1>       jz     short csftdf2_print_percent_1
3734 0000C62F 4F                       <1>       dec    edi
3735                                    <1>       ;cbw
3736 0000C630 28E4                     <1>       sub    ah, ah ; 09/12/2017
3737 0000C632 F6F1                     <1>       div    cl
3738 0000C634 80C430                   <1>       add    ah, '0'
3739 0000C637 8827                     <1>       mov    [edi], ah
3740                                    <1>       ;and   al, al
3741                                    <1>       ;jz    short csftdf2_print_percent_1
3742                                    <1>       ;dec   edi
3743                                    <1>       ;mov   [edi], '1' ; 100%
3744                                    <1>
3745                                    <1> csftdf2_print_percent_1:
3746 0000C639 BE[0D470100]             <1>       mov    esi, percentagestr
3747                                    <1>       ;call  print_msg
3748                                    <1>       ;retn
3749 0000C63E E922AFFFFF               <1>       jmp    print_msg
3750                                    <1>
3751                                    <1> csftdf2_read_file_sectors:
3752                                    <1>       ; 19/03/2016
3753 0000C643 807E0300                 <1>       cmp    byte [esi+LD_FATType], 0
3754 0000C647 0F8627070000             <1>         jna     csftdf2_read_fs_file_sectors
3755                                    <1>
3756                                    <1> csftdf2_read_fat_file_sectors:
3757                                    <1>       ; 19/03/2016
3758                                    <1>       ; 18/03/2016
3759                                    <1>       ; return:
3760                                    <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
3761                                    <1>       ;   CF = 0 & EDX = 0 -> not EOF
3762                                    <1>       ;   CF = 1 -> read error (error code in AL)
3763                                    <1>
3764                                    <1> csftdf2_read_fat_file_secs_0:
3765 0000C64D 8B15[F0950100]           <1>       mov    edx, [csftdf_filesize]
3766 0000C653 2B15[0C960100]           <1>       sub    edx, [csftdf_sf_rbytes]
3767 0000C659 3B15[04960100]           <1>       cmp    edx, [csftdf_r_size]
3768 0000C65F 7306                     <1>       jnb    short csftdf2_read_fat_file_secs_1
3769 0000C661 8915[04960100]           <1>       mov    [csftdf_r_size], edx
3770                                    <1>
3771                                    <1> csftdf2_read_fat_file_secs_1:
3772 0000C667 A1[04960100]             <1>       mov    eax, [csftdf_r_size]
3773 0000C66C 29D2                     <1>       sub    edx, edx
3774 0000C66E 0FB74E11                 <1>       movzx  ecx, word [esi+LD_BPB+BytesPerSec]
3775 0000C672 01C8                     <1>       add    eax, ecx
3776 0000C674 48                       <1>       dec    eax
3777 0000C675 F7F1                     <1>       div    ecx
3778 0000C677 89C1                     <1>       mov    ecx, eax ; sector count
3779 0000C679 A1[FC950100]             <1>       mov    eax, [csftdf_sf_cluster]
3780                                    <1>
3781                                    <1>       ; EBX = memory block address (current)
3782                                    <1>
3783 0000C67E E821090000               <1>       call   read_fat_file_sectors
3784 0000C683 7235                     <1>       jc     short csftdf2_read_fat_file_secs_3
3785                                    <1>
3786                                    <1>       ; EBX = next memory address
3787                                    <1>
3788 0000C685 A1[0C960100]             <1>       mov    eax, [csftdf_sf_rbytes]
3789 0000C68A 0305[04960100]           <1>       add    eax, [csftdf_r_size]
3790 0000C690 8B15[F0950100]           <1>       mov    edx, [csftdf_filesize]
3791 0000C696 39D0                     <1>       cmp    eax, edx
3792 0000C698 7320                     <1>       jnb    short csftdf2_read_fat_file_secs_3 ; edx > 0
3793 0000C69A A3[0C960100]             <1>       mov    [csftdf_sf_rbytes], eax
3794                                    <1>
3795 0000C69F 53                       <1>       push   ebx ; *
3796                                    <1>       ; get next cluster (csftdf_r_size! bytes)
3797 0000C6A0 A1[FC950100]             <1>       mov    eax, [csftdf_sf_cluster]
3798 0000C6A5 E8CC060000               <1>       call   get_next_cluster
```

```
3799 0000C6AA 5B                <1>       pop    ebx ; *
3800 0000C6AB 7306              <1>       jnc    short csftdf2_read_fat_file_secs_2
3801                            <1>
3802                            <1>       ; 15/10/2016
3803                            <1>       ;Disk read error instad of drv not ready err
3804 0000C6AD B811000000        <1>       mov    eax, 17 ; Read error !
3805 0000C6B2 C3                <1>       retn
3806                            <1>
3807                            <1> csftdf2_read_fat_file_secs_2:
3808 0000C6B3 29D2              <1>       sub    edx, edx ; 0
3809 0000C6B5 A3[FC950100]      <1>       mov    [csftdf_sf_cluster], eax ; next cluster
3810                            <1>
3811                            <1> csftdf2_read_fat_file_secs_3:
3812 0000C6BA C3                <1>       retn
3813                            <1>
3814                            <1> csftdf2_read_sf_cluster:
3815                            <1>       ; 19/03/2016
3816 0000C6BB BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
3817                            <1>
3818 0000C6C0 803D[14960100]00  <1>       cmp    byte [csftdf_percentage], 0
3819 0000C6C7 760D              <1>       jna    short csftdf2_read_sf_clust_2
3820                            <1>
3821 0000C6C9 53                <1>       push   ebx ; *
3822                            <1>
3823                            <1> csftdf2_read_sf_clust_next:
3824 0000C6CA E836FFFFFF        <1>       call   csftdf2_print_percentage
3825                            <1>
3826                            <1> csftdf2_read_sf_clust_0:
3827 0000C6CF 8B35[18960100]    <1>       mov    esi, [csftdf_sf_drv_dt]
3828                            <1> csftdf2_read_sf_clust_1:
3829 0000C6D5 5B                <1>       pop    ebx ; *
3830                            <1>
3831                            <1> csftdf2_read_sf_clust_2:
3832 0000C6D6 89DA              <1>       mov    edx, ebx
3833 0000C6D8 0315[04960100]    <1>       add    edx, [csftdf_r_size]
3834 0000C6DE 81FA00000800      <1>       cmp    edx, Cluster_Buffer + 65536
3835 0000C6E4 772F              <1>       ja     short csftdf2_write_df_cluster
3836                            <1>
3837 0000C6E6 E858FFFFFF        <1>       call   csftdf2_read_file_sectors ; 19/03/2016
3838 0000C6EB 0F8280020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
3839                            <1>
3840 0000C6F1 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
3841 0000C6F3 7520              <1>       jnz    short csftdf2_write_df_cluster
3842                            <1>
3843 0000C6F5 803D[14960100]00  <1>       cmp    byte [csftdf_percentage], 0
3844 0000C6FC 76D8              <1>       jna    short csftdf2_read_sf_clust_2
3845                            <1>
3846 0000C6FE 53                <1>       push   ebx ; *
3847                            <1>
3848                            <1>       ; Set cursor position
3849                            <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
3850 0000C6FF 8A3D[15960100]    <1>       mov    bh, [csftdf_videopage]
3851 0000C705 668B15[16960100]  <1>       mov    dx, [csftdf_cursorpos]
3852 0000C70C B402              <1>       mov    ah, 2
3853 0000C70E E85E50FFFF        <1>       call   _int10h
3854 0000C713 EBB5              <1>       jmp    short csftdf2_read_sf_clust_next
3855                            <1>
3856                            <1> csftdf2_write_df_cluster:
3857                            <1>       ; 19/03/2016
3858 0000C715 8B35[1C960100]    <1>       mov    esi, [csftdf_df_drv_dt]
3859 0000C71B BB00000700        <1>       mov    ebx, Cluster_Buffer ; buffer address (64KB)
3860                            <1>
3861                            <1> csftdf2_write_df_clust_next:
3862 0000C720 E855000000        <1>       call   csftdf2_write_file_sectors ; 19/03/2016
3863 0000C725 0F8246020000      <1>         jc     csftdf2_save_fat_file_err2 ; eocc! or disk error!
3864                            <1>
3865 0000C72B 09D2              <1>       or     edx, edx ; edx > 0 -> EOF
3866 0000C72D 750A              <1>       jnz    short csftdf2_rw_f_clust_ok
3867                            <1>
3868 0000C72F 81FB00000800      <1>       cmp    ebx, Cluster_Buffer + 65536
3869 0000C735 72E9              <1>       jb     short csftdf2_write_df_clust_next
3870                            <1>
3871 0000C737 EB82              <1>       jmp    short csftdf2_read_sf_cluster
3872                            <1>
3873                            <1> csftdf2_rw_f_clust_ok:
3874 0000C739 803D[14960100]00  <1>       cmp    byte [csftdf_percentage], 0
3875 0000C740 0F86B2010000      <1>         jna    csftdf2_save_fat_file_4 ; 25/03/2016
3876                            <1>
3877                            <1>       ; "100%"
3878 0000C746 BF[0D470100]      <1>       mov    edi, percentagestr
3879 0000C74B B031              <1>       mov    al, '1'
3880 0000C74D AA                <1>       stosb
3881 0000C74E B030              <1>       mov    al, '0'
3882 0000C750 AA                <1>       stosb
3883 0000C751 AA                <1>       stosb
3884                            <1>
3885 0000C752 8A3D[15960100]    <1>       mov    bh, [csftdf_videopage]
3886 0000C758 668B15[16960100]  <1>       mov    dx, [csftdf_cursorpos]
3887 0000C75F B402              <1>       mov    ah, 2
3888 0000C761 E80B50FFFF        <1>       call   _int10h
3889                            <1>
3890 0000C766 BE[0D470100]      <1>       mov    esi, percentagestr
3891 0000C76B E8F5ADFFFF        <1>       call   print_msg
3892                            <1>
3893 0000C770 E983010000        <1>         jmp    csftdf2_save_fat_file_4
3894                            <1>
3895                            <1> csftdf2_load_fs_file:
3896                            <1>       ; temporary - 18/03/2016
3897 0000C775 E96F020000        <1>         jmp    csftdf2_read_error
3898                            <1>
3899                            <1> csftdf2_write_file_sectors:
3900                            <1>       ; 19/03/2016
3901 0000C77A 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
3902 0000C77E 0F86F1050000      <1>         jna    csftdf2_write_fs_file_sectors
3903                            <1>
```

```
                                <1> csftdf2_write_fat_file_sectors:
                                <1>       ; 19/03/2016
                                <1>       ; 18/03/2016
                                <1>       ; return:
                                <1>       ;   CF = 0 & EDX > 0 -> END OF FILE
                                <1>       ;   CF = 0 & EDX = 0 -> not EOF
                                <1>       ;   CF = 1 -> write error (error code in AL)
                                <1>
                                <1> csftdf2_write_fat_file_secs_0:
0000C784 8B15[F0950100]         <1>       mov   edx, [csftdf_filesize]
0000C78A 2B15[10960100]         <1>       sub   edx, [csftdf_df_wbytes]
0000C790 3B15[08960100]         <1>       cmp   edx, [csftdf_w_size]
0000C796 7306                   <1>       jnb   short csftdf2_write_fat_file_secs_1
0000C798 8915[08960100]         <1>       mov   [csftdf_w_size], edx
                                <1>
                                <1> csftdf2_write_fat_file_secs_1:
0000C79E A1[08960100]           <1>       mov   eax, [csftdf_w_size]
0000C7A3 29D2                   <1>       sub   edx, edx
0000C7A5 0FB74E11               <1>       movzx ecx, word [esi+LD_BPB+BytesPerSec]
0000C7A9 01C8                   <1>       add   eax, ecx
0000C7AB 48                     <1>       dec   eax
0000C7AC F7F1                   <1>       div   ecx
0000C7AE 89C1                   <1>       mov   ecx, eax ; sector count
0000C7B0 A1[00960100]           <1>       mov   eax, [csftdf_df_cluster]
                                <1>
                                <1>       ; EBX = memory block address (current)
                                <1>
0000C7B5 E8A20F0000             <1>       call  write_fat_file_sectors
0000C7BA 7259                   <1>       jc    short csftdf2_write_fat_file_secs_4
                                <1>
                                <1>       ; EBX = next memory address
                                <1>
0000C7BC A1[10960100]           <1>       mov   eax, [csftdf_df_wbytes]
0000C7C1 0305[08960100]         <1>       add   eax, [csftdf_w_size]
0000C7C7 8B15[F0950100]         <1>       mov   edx, [csftdf_filesize]
0000C7CD 39D0                   <1>       cmp   eax, edx
0000C7CF 7344                   <1>       jnb   short csftdf2_write_fat_file_secs_4
0000C7D1 A3[10960100]           <1>       mov   [csftdf_df_wbytes], eax
                                <1>       ;
0000C7D6 A3[B2950100]           <1>       mov   [DestinationFile_DirEntry+DirEntry_FileSize], eax
                                <1>
0000C7DB 53                     <1>       push  ebx ; *
                                <1>
0000C7DC 803D[EE950100]01       <1>       cmp   byte [DestinationFileFound], 1
0000C7E3 7210                   <1>       jb    short csftdf2_write_fat_file_secs_2
                                <1>
                                <1>       ; get next cluster (csftdf_w_size! bytes)
0000C7E5 A1[00960100]           <1>       mov   eax, [csftdf_df_cluster]
0000C7EA E887050000             <1>       call  get_next_cluster
0000C7EF 731C                   <1>       jnc   short csftdf2_write_fat_file_secs_3
                                <1>
0000C7F1 21C0                   <1>       and   eax, eax ; end of cluster chain!?
0000C7F3 7521                   <1>       jnz   short csftdf2_write_fat_file_secs_5 ; disk error !
                                <1>
                                <1> csftdf2_write_fat_file_secs_2:
0000C7F5 A1[00960100]           <1>       mov   eax, [csftdf_df_cluster] ; last cluster
0000C7FA E8800E0000             <1>       call  add_new_cluster
0000C7FF 7215                   <1>       jc    short csftdf2_write_fat_file_secs_5
                                <1>
                                <1>       ; NOTE: Destination file size may be bigger than
                                <1>       ; source file size when the last reading fails after here.
                                <1>       ; (The last -empty- cluster of destination file must be
                                <1>       ; truncated and LMDT must be current date&time for partial
                                <1>       ; copy result!)
0000C801 8B15[08960100]         <1>       mov   edx, [csftdf_w_size] ; bytes per cluster
0000C807 0115[B2950100]         <1>       add   [DestinationFile_DirEntry+DirEntry_FileSize], edx
                                <1>
                                <1> csftdf2_write_fat_file_secs_3:
0000C80D 5B                     <1>       pop   ebx ; *
0000C80E 29D2                   <1>       sub   edx, edx ; 0
0000C810 A3[00960100]           <1>       mov   [csftdf_df_cluster], eax ; next cluster
                                <1>
                                <1> csftdf2_write_fat_file_secs_4:
0000C815 C3                     <1>       retn
                                <1>
                                <1> csftdf2_write_fat_file_secs_5:
0000C816 5B                     <1>       pop   ebx ; *
                                <1>       ; 16/10/2016 (1Dh -> 18)
0000C817 B812000000             <1>       mov   eax, 18 ; Write error !
0000C81C C3                     <1>       retn
                                <1>
                                <1> csftdf2_save_file:
                                <1>       ; 09/12/2017
                                <1>       ; 25/03/2016
                                <1>       ; 19/03/2016
                                <1>       ; 18/03/2016
0000C81D 8B35[1C960100]         <1>       mov   esi, [csftdf_df_drv_dt] ; logical dos drv desc. tbl.
                                <1>
0000C823 8B1D[F4950100]         <1>       mov   ebx, [csftdf_sf_mem_addr] ; memory block address
                                <1>
0000C829 807E0300               <1>       cmp   byte [esi+LD_FATType], 0
0000C82D 0F86F4010000           <1>         jna     csftdf2_save_fs_file
                                <1>
                                <1> csftdf2_save_fat_file:
0000C833 53                     <1>       push  ebx; *
                                <1>
0000C834 803D[14960100]00       <1>       cmp   byte [csftdf_percentage], 0
0000C83B 7724                   <1>       ja    short csftdf2_save_fat_file_0
                                <1>
                                <1>       ; Set cursor position
                                <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
0000C83D 8A3D[15960100]         <1>       mov   bh, [csftdf_videopage]
0000C843 668B15[16960100]       <1>       mov   dx, [csftdf_cursorpos]
0000C84A B402                   <1>       mov   ah, 2
0000C84C E8204FFFFF             <1>       call  _int10h
```

```
4009                                    <1>
4010 0000C851 BE[01470100]              <1>       mov   esi, msg_writing
4011 0000C856 E80AADFFFF                <1>       call  print_msg
4012                                    <1>
4013                                    <1> csftdf2_save_fat_file_next:
4014 0000C85B 8B35[1C960100]            <1>       mov   esi, [csftdf_df_drv_dt] ; 25/03/2016
4015                                    <1>
4016                                    <1> csftdf2_save_fat_file_0:
4017 0000C861 5B                        <1>       pop   ebx ; *
4018                                    <1>
4019                                    <1> csftdf2_save_fat_file_1:
4020 0000C862 E813FFFFFF                <1>       call  csftdf2_write_file_sectors ; 19/03/2016
4021 0000C867 0F827E010000              <1>         jc      csftdf2_rw_error ; eocc! or disk error!
4022                                    <1>
4023 0000C86D 09D2                      <1>       or    edx, edx ; edx > 0 -> EOF
4024 0000C86F 756D                      <1>         jnz     short csftdf2_save_fat_file_3 ; 25/03/2016
4025                                    <1>
4026 0000C871 803D[14960100]00          <1>       cmp   byte [csftdf_percentage], 0
4027 0000C878 76E8                      <1>       jna   short csftdf2_save_fat_file_1
4028                                    <1>
4029 0000C87A B020                      <1>       mov   al, 20h
4030 0000C87C BF[0D470100]              <1>       mov   edi, percentagestr
4031 0000C881 AA                        <1>       stosb
4032 0000C882 AA                        <1>       stosb
4033 0000C883 A1[10960100]              <1>       mov   eax, [csftdf_df_wbytes]
4034 0000C888 BA64000000                <1>       mov   edx, 100
4035 0000C88D F7E2                      <1>       mul   edx
4036 0000C88F 8B0D[F0950100]            <1>       mov   ecx, [csftdf_filesize]
4037 0000C895 F7F1                      <1>       div   ecx
4038 0000C897 B10A                      <1>       mov   cl, 10
4039 0000C899 F6F1                      <1>       div   cl
4040 0000C89B 80C430                    <1>       add   ah, '0'
4041 0000C89E 8827                      <1>       mov   [edi], ah
4042 0000C8A0 20C0                      <1>       and   al, al
4043 0000C8A2 740A                      <1>       jz    short csftdf2_save_fat_file_2
4044 0000C8A4 4F                        <1>       dec   edi
4045                                    <1>       ;cbw
4046 0000C8A5 30E4                      <1>       xor   ah, ah ; 09/12/2017
4047 0000C8A7 F6F1                      <1>       div   cl
4048 0000C8A9 80C430                    <1>       add   ah, '0'
4049 0000C8AC 8827                      <1>       mov   [edi], ah
4050                                    <1>       ;and  al, al
4051                                    <1>       ;jz   short csftdf2_save_fat_file_2
4052                                    <1>       ;dec  edi
4053                                    <1>       ;mov  [edi], '1' ; 100%
4054                                    <1>
4055                                    <1> csftdf2_save_fat_file_2:
4056 0000C8AE 53                        <1>       push  ebx ; *
4057                                    <1>
4058 0000C8AF E802000000                <1>       call  csftdf2_print_wr_percentage ; 25/03/2016
4059                                    <1>
4060 0000C8B4 EBA5                      <1>         jmp     csftdf2_save_fat_file_next
4061                                    <1>
4062                                    <1> csftdf2_print_wr_percentage:
4063                                    <1>       ; Set cursor position
4064                                    <1>       ; AH= 02h, BH= Page Number, DH= Row, DL= Column
4065 0000C8B6 8A3D[15960100]            <1>       mov   bh, [csftdf_videopage]
4066 0000C8BC 668B15[16960100]          <1>       mov   dx, [csftdf_cursorpos]
4067 0000C8C3 B402                      <1>       mov   ah, 2
4068 0000C8C5 E8A74EFFFF                <1>       call  _int10h
4069                                    <1>
4070 0000C8CA BE[01470100]              <1>       mov   esi, msg_writing
4071 0000C8CF E891ACFFFF                <1>       call  print_msg
4072                                    <1>
4073 0000C8D4 BE[0D470100]              <1>       mov   esi, percentagestr
4074                                    <1>       ;call print_msg
4075                                    <1>       ;retn
4076 0000C8D9 E987ACFFFF                <1>       jmp   print_msg
4077                                    <1>
4078                                    <1> csftdf2_save_fat_file_3:
4079 0000C8DE 803D[14960100]00          <1>       cmp   byte [csftdf_percentage], 0
4080 0000C8E5 7611                      <1>         jna     csftdf2_save_fat_file_4 ; 25/03/2016
4081                                    <1>
4082                                    <1>       ; "100%"
4083 0000C8E7 BF[0D470100]              <1>       mov   edi, percentagestr
4084 0000C8EC B031                      <1>       mov   al, '1'
4085 0000C8EE AA                        <1>       stosb
4086 0000C8EF B030                      <1>       mov   al, '0'
4087 0000C8F1 AA                        <1>       stosb
4088 0000C8F2 AA                        <1>       stosb
4089                                    <1>
4090 0000C8F3 E8BEFFFFFF                <1>       call  csftdf2_print_wr_percentage
4091                                    <1>
4092                                    <1> csftdf2_save_fat_file_4:
4093 0000C8F8 803D[EE950100]00          <1>       cmp   byte [DestinationFileFound], 0
4094 0000C8FF 7647                      <1>       jna   short csftdf2_save_fat_file_6
4095                                    <1>
4096 0000C901 8B35[1C960100]            <1>       mov   esi, [csftdf_df_drv_dt] ; 31/03/2016
4097                                    <1>
4098 0000C907 A1[00960100]              <1>       mov   eax, [csftdf_df_cluster] ; last cluster
4099 0000C90C E865040000                <1>       call  get_next_cluster
4100 0000C911 7235                      <1>       jc    short csftdf2_save_fat_file_6 ; eocc! or disk error!
4101                                    <1>
4102 0000C913 A1[00960100]              <1>       mov   eax, [csftdf_df_cluster] ; last cluster
4103                                    <1>       ;xor  ecx, ecx
4104                                    <1>       ;mov  [FAT_ClusterCounter], ecx ; 0 ; reset
4105                                    <1>       ;dec  ecx ; 0FFFFFFFFh
4106                                    <1>       ;shr  ecx, 4 ; 28 bit ; 0FFFFFFFh
4107 0000C918 B9FFFFFF0F                <1>       mov   ecx, 0FFFFFFFh
4108 0000C91D E87E070000                <1>       call  update_cluster
4109 0000C922 7224                      <1>       jc    short csftdf2_save_fat_file_6 ; really last cluster!?
4110                                    <1>
4111 0000C924 A3[00960100]              <1>       mov   [csftdf_df_cluster], eax ; next cluster
4112                                    <1>
4113                                    <1>       ; byte [FAT_BuffValidData] = 2
```

```
4114 0000C929 E82F0A0000          <1>       call   save_fat_buffer
4115 0000C92E 730E                <1>       jnc    short csftdf2_save_fat_file_5
4116                              <1>
4117 0000C930 8B15[F0950100]      <1>       mov    edx, [csftdf_filesize]
4118 0000C936 8915[B2950100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
4119 0000C93C EB58                <1>       jmp    short csftdf2_save_fat_file_err3
4120                              <1>
4121                              <1> csftdf2_save_fat_file_5:
4122 0000C93E A1[00960100]        <1>       mov    eax, [csftdf_df_cluster]
4123                              <1>
4124                              <1>       ; EAX = First cluster to be truncated/unlinked
4125                              <1>       ; ESI = Logical dos drive description table address
4126 0000C943 E8580C0000          <1>       call   truncate_cluster_chain
4127                              <1>
4128                              <1> csftdf2_save_fat_file_6:
4129                              <1>       ; 28/03/2016
4130 0000C948 BE[21950100]        <1>       mov    esi, SourceFile_DirEntry+DirEntry_Attr ; +11 to + 18
4131 0000C94D BF[A1950100]        <1>       mov    edi, DestinationFile_DirEntry+DirEntry_Attr ; +11 to + 18
4132 0000C952 A4                  <1>       movsb ; +11
4133 0000C953 A5                  <1>       movsd ; +12 .. +15
4134 0000C954 66A5                <1>       movsw ; +16 .. +17
4135                              <1>             ; + 18
4136 0000C956 83C604              <1>       add    esi, 4
4137 0000C959 83C704              <1>       add    edi, 4
4138 0000C95C A5                  <1>       movsd  ; DirEntry_WrtTime ; +22 .. +25
4139                              <1>
4140 0000C95D 8B15[F0950100]      <1>       mov    edx, [csftdf_filesize]
4141 0000C963 8915[B2950100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], edx
4142                              <1>
4143 0000C969 E8BAF0FFFF          <1>       call   convert_current_date_time
4144                              <1>       ; DX = Date in dos dir entry format
4145                              <1>       ; AX = Time in dos dir entry format
4146 0000C96E EB4D                <1>       jmp    short csftdf2_save_fat_file_7
4147                              <1>
4148                              <1> csftdf2_save_fat_file_err1:
4149 0000C970 5B                  <1>       pop    ebx ; *
4150                              <1> csftdf2_save_fat_file_err2:
4151 0000C971 A1[10960100]        <1>       mov    eax, [csftdf_df_wbytes]
4152 0000C976 8B15[B2950100]      <1>       mov    edx, [DestinationFile_DirEntry+DirEntry_FileSize]
4153 0000C97C 39C2                <1>       cmp    edx, eax
4154 0000C97E 7616                <1>       jna    short csftdf2_save_fat_file_err3
4155 0000C980 A1[00960100]        <1>       mov    eax, [csftdf_df_cluster] ; last (empty) cluster
4156                              <1>       ; ESI = Logical dos drive description table address
4157 0000C985 E8160C0000          <1>       call   truncate_cluster_chain
4158 0000C98A 720A                <1>       jc     short csftdf2_save_fat_file_err3
4159 0000C98C A1[10960100]        <1>       mov    eax, [csftdf_df_wbytes]
4160 0000C991 A3[B2950100]        <1>       mov    [DestinationFile_DirEntry+DirEntry_FileSize], eax
4161                              <1> csftdf2_save_fat_file_err3:
4162 0000C996 E88DF0FFFF          <1>       call   convert_current_date_time
4163                              <1>       ; DX = Date in dos dir entry format
4164                              <1>       ; AX = Time in dos dir entry format
4165 0000C99B C605[A3950100]00    <1>       mov    byte [DestinationFile_DirEntry+DirEntry_CrtTimeTenth], 0
4166 0000C9A2 66A3[A4950100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtTime], ax
4167 0000C9A8 668915[A6950100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_CrtDate], dx
4168 0000C9AF 66A3[AC950100]      <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtTime], ax
4169 0000C9B5 668915[AE950100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_WrtDate], dx
4170 0000C9BC F9                  <1>       stc
4171                              <1> csftdf2_save_fat_file_7:
4172 0000C9BD 9C                  <1>       pushf
4173 0000C9BE 668915[A8950100]    <1>       mov    [DestinationFile_DirEntry+DirEntry_LastAccDate], dx
4174 0000C9C5 BE[96950100]        <1>       mov    esi, DestinationFile_DirEntry
4175 0000C9CA BF00000800          <1>       mov    edi, Directory_Buffer
4176 0000C9CF 0FB70D[BE950100]    <1>       movzx  ecx, word [DestinationFile_DirEntryNumber] ; (<2048)
4177 0000C9D6 66C1E105            <1>       shl    cx, 5 ; 32 * directory entry number
4178 0000C9DA 01CF                <1>       add    edi, ecx
4179                              <1>       ;mov   ecx, 8
4180 0000C9DC 66B90800            <1>       mov    cx, 8
4181 0000C9E0 F3A5                <1>       rep    movsd
4182 0000C9E2 9D                  <1>       popf
4183 0000C9E3 730B                <1>       jnc    short csftdf2_write_file_OK
4184                              <1>
4185                              <1> csftdf2_write_error:
4186                              <1>       ; 18/03/2016
4187 0000C9E5 B01D                <1>       mov    al, 1Dh ; write error
4188 0000C9E7 EB02                <1>       jmp    short csftdf2_rw_error
4189                              <1>
4190                              <1>       ; 16/03/2016
4191                              <1> csftdf2_read_error:
4192 0000C9E9 B011                <1>       mov    al, 17 ; ; Drive not ready or read error!
4193                              <1> csftdf2_rw_error:
4194 0000C9EB A2[ED950100]        <1>       mov    [csftdf_rw_err], al
4195                              <1>
4196                              <1> csftdf2_write_file_OK:
4197                              <1>       ; 18/03/2016
4198 0000C9F0 C605[04920100]02    <1>       mov    byte [DirBuff_ValidData], 2
4199 0000C9F7 E8CAF0FFFF          <1>       call   save_directory_buffer
4200                              <1>
4201                              <1>       ; Update last modification date&time of destination
4202                              <1>       ; file's (parent) directory
4203 0000C9FC E860F1FFFF          <1>       call   update_parent_dir_lmdt
4204                              <1>       ;
4205 0000CA01 A1[F4950100]        <1>       mov    eax, [csftdf_sf_mem_addr] ; start address
4206                              <1>
4207 0000CA06 21C0                <1>       and    eax, eax
4208 0000CA08 750E                <1>       jnz    short csftdf2_dealloc_mblock
4209                              <1>
4210 0000CA0A 88C5                <1>       mov    ch, al ; 0 (Cluster r/w, not full loading)
4211                              <1> csftdf2_dealloc_retn:
4212 0000CA0C 8A0D[ED950100]      <1>       mov    cl, [csftdf_rw_err]
4213 0000CA12 A1[00960100]        <1>       mov    eax, [csftdf_df_cluster]
4214 0000CA17 C3                  <1>       retn
4215                              <1>
4216                              <1> csftdf2_dealloc_mblock:
4217 0000CA18 8B0D[F8950100]      <1>       mov    ecx, [csftdf_sf_mem_bsize] ; block size
4218 0000CA1E E86F9CFFFF          <1>       call   deallocate_memory_block
```

```
4219 0000CA23 B5FF                <1>         mov     ch, 0FFh ; (File was full loaded at memory)
4220 0000CA25 EBE5                <1>       jmp    short csftdf2_dealloc_retn
4221                              <1>
4222                              <1> csftdf2_save_fs_file:
4223                              <1>       ; 16/10/2016 (1Dh -> 18)
4224                              <1>       ; temporary - (21/03/2016)
4225 0000CA27 B812000000          <1>       mov    eax, 18 ; write error
4226 0000CA2C F9                  <1>       stc
4227 0000CA2D C3                  <1>       retn
4228                              <1>
4229                              <1> create_file:
4230                              <1>       ; 16/10/2016
4231                              <1>       ; 24/03/2016, 31/03/2016
4232                              <1>       ; 20/03/2016, 21/03/2016, 23/03/2016
4233                              <1>       ; 19/03/2016 (TRDOS 396 = TRDOS v2.0)
4234                              <1>       ; 03/09/2011 (FILE.ASM, 'proc_create_file')
4235                              <1>       ; 09/08/2010
4236                              <1>       ;
4237                              <1>       ; INPUT ->
4238                              <1>       ;      EAX = File Size
4239                              <1>       ;      ESI = ASCIIZ File Name
4240                              <1>       ;      CL = File Attributes
4241                              <1>       ;      EBX = FFFFFFFFh -> create empty file
4242                              <1>       ;                         (only for FAT fs)
4243                              <1>       ; OUTPUT ->
4244                              <1>       ;     CF = 0 ->
4245                              <1>       ;      EAX = New file's first cluster
4246                              <1>       ;      ESI = Logical Dos Drv Descr. Table Addr.
4247                              <1>       ;      EBX = offset CreateFile_Size
4248                              <1>       ;      ECX = Sectors per cluster (<256)
4249                              <1>       ;      EDX = Directory entry index/number (<65536)
4250                              <1>       ;     CF = 1 -> error code in AL
4251                              <1>
4252                              <1> ;     test   cl, 18h (directory or volume name)
4253                              <1> ;     jnz    short loc_createfile_access_denied
4254 0000CA2E 80E107              <1>       and    cl, 07h ; S, H, R
4255 0000CA31 880D[3C960100]      <1>         mov     [createfile_attrib], cl
4256                              <1>
4257 0000CA37 89D9                <1>       mov    ecx, ebx
4258 0000CA39 89F3                <1>       mov    ebx, esi ; ASCIIZ File Name address
4259 0000CA3B 29D2                <1>       sub    edx, edx
4260 0000CA3D 8A35[DE8A0100]      <1>         mov     dh, [Current_Drv]
4261 0000CA43 BE00010900          <1>         mov     esi, Logical_DOSDisks
4262 0000CA48 01D6                <1>       add    esi, edx
4263                              <1>
4264 0000CA4A 8815[47960100]      <1>       mov    [createfile_UpdatePDir], dl ; 0 ; 31/03/2016
4265                              <1>
4266                              <1>       ; LD_DiskType = 0 for write protection (read only)
4267 0000CA50 807E0101            <1>       cmp    byte [esi+LD_DiskType], 1 ; 0 = Invalid
4268 0000CA54 730A                <1>       jnb    short loc_createfile_check_file_sytem
4269                              <1>       ; 16/10/2016 (TRDOS Error code: 30, disk write protected)
4270 0000CA56 B81E000000          <1>       mov    eax, 30 ; 13h, MSDOS err : Disk write-protected
4271 0000CA5B 66BA0000            <1>       mov    dx, 0
4272                              <1>       ; err retn: EDX = 0, EBX = File name offset
4273                              <1>       ; ESI -> Dos drive description table address
4274 0000CA5F C3                  <1>       retn
4275                              <1>
4276                              <1> ;loc_createfile_access_denied:
4277                              <1> ;     mov    eax, 05h ; access denied (invalid attributes input)
4278                              <1> ;     stc
4279                              <1> ;     retn
4280                              <1>
4281                              <1> loc_createfile_check_file_sytem:
4282 0000CA60 807E0301            <1>       cmp    byte [esi+LD_FATType], 1
4283 0000CA64 730A                <1>       jnb    short loc_createfile_chk_empty_FAT_file_sign1
4284                              <1>
4285 0000CA66 A3[28960100]        <1>       mov    [createfile_size], eax
4286                              <1>       ; ESI = Logical Dos Drive Description Table address
4287                              <1>       ; EBX = ASCIIZ File Name address
4288 0000CA6B E9FE020000          <1>       jmp    create_fs_file
4289                              <1>
4290                              <1> loc_createfile_chk_empty_FAT_file_sign1:
4291                              <1>       ; ECX = FFFFFFFFh -> create empty file if drive has FAT fs
4292 0000CA70 41                  <1>       inc    ecx
4293 0000CA71 7506                <1>       jnz    short loc_createfile_chk_empty_FAT_file_sign2
4294 0000CA73 890D[28960100]      <1>       mov    [createfile_size], ecx ; 0 ; empty file
4295                              <1>
4296                              <1> loc_createfile_chk_empty_FAT_file_sign2:
4297                              <1>       ; 23/03/2016
4298 0000CA79 668B4E11            <1>       mov    cx, [esi+LD_BPB+BytesPerSec]
4299 0000CA7D 66890D[44960100]    <1>       mov    [createfile_BytesPerSec], cx
4300                              <1>
4301                              <1>       ; EBX = ASCIIZ File Name address
4302 0000CA84 0FB65613            <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
4303 0000CA88 8815[3D960100]      <1>       mov    [createfile_SecPerClust], dl
4304 0000CA8E 8B4E74              <1>       mov    ecx, [esi+LD_FreeSectors]
4305 0000CA91 39D1                <1>       cmp    ecx, edx ; byte [createfile_SecPerClust]
4306 0000CA93 7306                <1>       jnb    short loc_create_fat_file
4307                              <1>
4308                              <1> loc_createfile_insufficient_disk_space:
4309 0000CA95 B827000000          <1>       mov    eax, 27h
4310                              <1> loc_createfile_gffc_retn:
4311 0000CA9A C3                  <1>       retn
4312                              <1>
4313                              <1> loc_create_fat_file:
4314 0000CA9B 891D[20960100]      <1>       mov    [createfile_Name_Offset], ebx
4315 0000CAA1 890D[24960100]      <1>       mov    [createfile_FreeSectors], ecx
4316                              <1>
4317                              <1> loc_createfile_gffc_1:
4318 0000CAA7 E821050000          <1>       call   get_first_free_cluster
4319 0000CAAC 72EC                <1>       jc     short loc_createfile_gffc_retn
4320                              <1>
4321 0000CAAE A3[2C960100]        <1>       mov    [createfile_FFCluster], eax
4322                              <1>
4323                              <1> loc_createfile_locate_ffe_on_directory:
```

```
4324                                    <1>       ; Current directory fcluster <> Directory buffer cluster
4325                                    <1>       ; Current directory will be reloaded by
4326                                    <1>       ; 'locate_current_dir_file' procedure
4327                                    <1>       ;
4328                                    <1>       ; ESI = Logical Dos Drv Desc. Table Adress
4329 0000CAB3 56                        <1>       push   esi ; *
4330 0000CAB4 31C0                      <1>       xor    eax, eax
4331                                    <1>
4332 0000CAB6 A3[FA910100]              <1>       mov    dword [FAT_ClusterCounter], eax ; 0
4333                                    <1>       ; 21/03/2016
4334 0000CABB A2[46960100]              <1>       mov    byte [createfile_wfc], al ; 0
4335                                    <1>
4336 0000CAC0 89C1                      <1>       mov    ecx, eax
4337 0000CAC2 6649                      <1>       dec    cx ; FFFFh
4338                                    <1>       ; CX = FFFFh -> find first deleted or free entry
4339                                    <1>       ; ESI would be ASCIIZ filename address if the call
4340                                    <1>       ; would not be for first free or deleted dir entry
4341 0000CAC4 E8D6E7FFFF                <1>       call   locate_current_dir_file
4342 0000CAC9 0F83EE000000              <1>       jnc    loc_createfile_set_ff_dir_entry
4343 0000CACF 5E                        <1>       pop    esi ; *
4344                                    <1>        ; ESI = Logical DOS Drv. Description Table Address
4345 0000CAD0 83F802                    <1>       cmp    eax, 2
4346 0000CAD3 7402                      <1>       je     short loc_createfile_add_new_cluster
4347                                    <1> loc_createfile_locate_file_stc_retn:
4348 0000CAD5 F9                        <1>       stc
4349 0000CAD6 C3                        <1>       retn
4350                                    <1>
4351                                    <1> loc_createfile_add_new_cluster:
4352 0000CAD7 803D[DD8A0100]02          <1>       cmp    byte [Current_FATType], 2
4353                                    <1>       ;cmp   byte [esi+LD_FATType], 2
4354 0000CADE 770C                      <1>       ja     short loc_createfile_add_new_cluster_check_fsc
4355 0000CAE0 803D[DC8A0100]01          <1>       cmp    byte [Current_Dir_Level], 1
4356                                    <1>       ;cmp   byte [esi+LD_CDirLevel], 1
4357 0000CAE7 7303                      <1>       jnb    short loc_createfile_add_new_cluster_check_fsc
4358                                    <1>
4359                                    <1>       ;mov   eax, 12
4360 0000CAE9 B00C                      <1>       mov    al, 12 ; No more files
4361                                    <1>
4362                                    <1> loc_createfile_anc_retn:
4363 0000CAEB C3                        <1>       retn
4364                                    <1>
4365                                    <1> loc_createfile_add_new_cluster_check_fsc:
4366 0000CAEC 8B0D[24960100]            <1>       mov    ecx, [createfile_FreeSectors]
4367 0000CAF2 0FB605[3D960100]          <1>       movzx  eax, byte [createfile_SecPerClust]
4368 0000CAF9 66D1E0                    <1>       shl    ax, 1 ; AX = 2 * AX
4369 0000CAFC 39C1                      <1>       cmp    ecx, eax
4370 0000CAFE 7295                      <1>         jb   short loc_createfile_insufficient_disk_space
4371                                    <1>
4372                                    <1> loc_createfile_add_new_subdir_cluster:
4373 0000CB00 8B15[09920100]            <1>       mov    edx, [DirBuff_Cluster]
4374 0000CB06 8915[30960100]            <1>       mov    [createfile_LastDirCluster], edx
4375                                    <1>
4376 0000CB0C A1[2C960100]              <1>       mov    eax, [createfile_FFCluster]
4377 0000CB11 E846040000                <1>       call   load_FAT_sub_directory
4378 0000CB16 72D3                      <1>       jc     short loc_createfile_anc_retn
4379                                    <1>
4380                                    <1> pass_createfile_add_new_subdir_cluster:
4381                                    <1>       ;movzx eax, word [esi+LD_BPB+BytesPerSec]
4382 0000CB18 0FB705[44960100]          <1>       movzx  eax, word [createfile_BytesPerSec] ; 23/03/2016
4383 0000CB1F F7E1                      <1>       mul    ecx ; ecx = directory buffer sector count
4384 0000CB21 89C1                      <1>       mov    ecx, eax
4385 0000CB23 C1E902                    <1>       shr    ecx, 2 ; dword count
4386 0000CB26 29C0                      <1>       sub    eax, eax ; 0
4387 0000CB28 F3AB                      <1>       rep    stosd
4388                                    <1>       ;
4389 0000CB2A C605[04920100]02          <1>       mov    byte [DirBuff_ValidData], 2
4390 0000CB31 E890EFFFFF                <1>       call   save_directory_buffer
4391 0000CB36 72B3                      <1>       jc     short loc_createfile_anc_retn
4392                                    <1>
4393                                    <1> loc_createfile_save_added_subdir_cluster:
4394 0000CB38 A1[30960100]              <1>       mov    eax, [createfile_LastDirCluster]
4395 0000CB3D 8B0D[2C960100]            <1>       mov    ecx, [createfile_FFCluster]
4396 0000CB43 E858050000                <1>       call   update_cluster
4397 0000CB48 7304                      <1>       jnc    short loc_createfile_save_fat_buffer_0
4398 0000CB4A 09C0                      <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4399 0000CB4C 751A                      <1>       jnz    short loc_createfile_save_fat_buffer_stc_retn
4400                                    <1>
4401                                    <1> loc_createfile_save_fat_buffer_0:
4402 0000CB4E A1[2C960100]              <1>       mov    eax, [createfile_FFCluster]
4403 0000CB53 A3[30960100]              <1>       mov    [createfile_LastDirCluster], eax
4404 0000CB58 B9FFFFFF0F                <1>       mov    ecx, 0FFFFFFFh ; 28 bit
4405 0000CB5D E83E050000                <1>       call   update_cluster
4406 0000CB62 7306                      <1>       jnc    short loc_createfile_save_fat_buffer_1
4407 0000CB64 09C0                      <1>       or     eax, eax ; Was it free cluster
4408 0000CB66 7402                      <1>       jz     short loc_createfile_save_fat_buffer_1
4409                                    <1>
4410                                    <1> loc_createfile_save_fat_buffer_stc_retn:
4411 0000CB68 F9                        <1>       stc
4412                                    <1> loc_createfile_save_fat_buffer_retn:
4413                                    <1> loc_createfile_gffc_2_stc_retn:
4414 0000CB69 C3                        <1>       retn
4415                                    <1>
4416                                    <1> loc_createfile_save_fat_buffer_1:
4417                                    <1>       ; byte [FAT_BuffValidData] = 2
4418 0000CB6A E8EE070000                <1>       call   save_fat_buffer
4419 0000CB6F 72F8                      <1>       jc     short loc_createfile_save_fat_buffer_retn
4420                                    <1>
4421 0000CB71 803D[FA910100]01          <1>       cmp    byte [FAT_ClusterCounter], 1
4422 0000CB78 7222                      <1>       jb     short loc_createfile_save_fat_buffer_2
4423                                    <1>
4424                                    <1>       ; ESI = Logical DOS Drive Description Table address
4425 0000CB7A A1[FA910100]              <1>       mov    eax, [FAT_ClusterCounter]
4426                                    <1>
4427 0000CB7F C605[FA910100]00          <1>       mov    byte [FAT_ClusterCounter], 0 ; 21/03/2016
4428                                    <1>
```

```
4429 0000CB86 66BB01FF          <1>       mov   bx, 0FF01h ; add free clusters
4430 0000CB8A E863080000        <1>       call  calculate_fat_freespace
4431                            <1>
4432                            <1>       ;inc  eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4433                            <1>       ;jnz  short loc_createfile_save_fat_buffer_2
4434                            <1>
4435                            <1>       ; ecx > 0 -> Recalculation is needed
4436 0000CB8F 09C9              <1>       or    ecx, ecx
4437 0000CB91 7409              <1>       jz    short loc_createfile_save_fat_buffer_2
4438                            <1>
4439 0000CB93 66BB00FF          <1>       mov   bx, 0FF00h ; ; recalculate free space
4440 0000CB97 E856080000        <1>       call  calculate_fat_freespace
4441                            <1>
4442                            <1> loc_createfile_save_fat_buffer_2:
4443                            <1>       ;call update_parent_dir_lmdt
4444                            <1>
4445                            <1> loc_createfile_gffc_2:
4446 0000CB9C E82C040000        <1>       call  get_first_free_cluster
4447 0000CBA1 72C6              <1>       jc    short loc_createfile_gffc_2_stc_retn
4448                            <1>
4449 0000CBA3 A3[2C960100]      <1>       mov   [createfile_FFCluster], eax
4450                            <1>
4451 0000CBA8 A1[30960100]      <1>       mov   eax, [createfile_LastDirCluster]
4452                            <1>
4453 0000CBAD E8AA030000        <1>       call  load_FAT_sub_directory
4454 0000CBB2 72B5              <1>       jc    short loc_createfile_gffc_2_stc_retn
4455                            <1>
4456 0000CBB4 BF00000800        <1>       mov   edi, Directory_Buffer
4457                            <1>
4458 0000CBB9 6629DB            <1>       sub   bx, bx ; directory entry index/number = 0
4459                            <1>
4460 0000CBBC 56                <1>       push  esi ; * ; 23/03/2016
4461                            <1>
4462                            <1> loc_createfile_set_ff_dir_entry:
4463 0000CBBD 66891D[3E960100]  <1>       mov   [createfile_DirIndex], bx
4464                            <1>
4465                            <1>         ; EDI = Directory entry address
4466 0000CBC4 8B35[20960100]    <1>       mov   esi, [createfile_Name_Offset]
4467 0000CBCA A1[2C960100]      <1>       mov   eax, [createfile_FFCluster]
4468 0000CBCF A3[34960100]      <1>       mov   [createfile_Cluster], eax ; 24/03/2016
4469 0000CBD4 B5FF              <1>       mov   ch, 0FFh
4470 0000CBD6 8A0D[3C960100]    <1>         mov cl, [createfile_attrib] ; file attributes
4471                            <1>       ; CH > 0 -> File size is in [EBX]
4472 0000CBDC BB[28960100]      <1>       mov   ebx, createfile_size
4473                            <1>
4474 0000CBE1 E803EEFFFF        <1>       call  make_directory_entry
4475                            <1>
4476 0000CBE6 5E                <1>       pop   esi ; * ; ESI = Logical Dos Drv Desc. Table address
4477                            <1>
4478 0000CBE7 C605[04920100]02  <1>       mov   byte [DirBuff_ValidData], 2
4479 0000CBEE E8D3EEFFFF        <1>       call  save_directory_buffer
4480 0000CBF3 7221              <1>       jc    short loc_createfile_set_ff_dir_entry_retn
4481                            <1>
4482 0000CBF5 C605[47960100]01  <1>       mov   byte [createfile_UpdatePDir], 1 ; 31/03/2016
4483                            <1>
4484                            <1> loc_createfile_get_set_write_file_cluster:
4485 0000CBFC A1[28960100]      <1>       mov   eax, [createfile_size]
4486 0000CC01 09C0              <1>       or    eax, eax
4487 0000CC03 7570              <1>       jnz   short loc_createfile_get_set_wfc_cont
4488 0000CC05 40                <1>       inc   eax
4489                            <1>       ; 23/03/2016
4490 0000CC06 0FB61D[3D960100]  <1>       movzx ebx, byte [createfile_SecPerClust]
4491                            <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
4492 0000CC0D 0FB70D[44960100]  <1>         movzx   ecx, word [createfile_BytesPerSec] ; 512
4493 0000CC14 EB7C              <1>       jmp   loc_createfile_set_cluster_count
4494                            <1>
4495                            <1> loc_createfile_set_ff_dir_entry_retn:
4496 0000CC16 C3                <1>       retn
4497                            <1>
4498                            <1> loc_createfile_write_fcluster_to_disk:
4499 0000CC17 034668            <1>       add   eax, [esi+LD_DATABegin] ; convert to physical address
4500 0000CC1A BB00000700        <1>       mov   ebx, Cluster_Buffer
4501                            <1>       ; ESI = Logical DOS Drv. Desc. Tbl. address
4502                            <1>       ; EAX = Disk address
4503                            <1>       ; EBX = Sector Buffer
4504                            <1>       ; ECX = sectors per cluster
4505 0000CC1F E86D5F0000        <1>       call  disk_write
4506 0000CC24 7211              <1>       jc    short loc_createfile_dsk_wr_err
4507                            <1>
4508                            <1> loc_createfile_update_fat_cluster:
4509                            <1>       ; 21/03/2016
4510 0000CC26 803D[46960100]00  <1>       cmp   byte [createfile_wfc], 0
4511 0000CC2D 7712              <1>       ja    short loc_createfile_update_fat_cluster_n1
4512                            <1>
4513 0000CC2F FE05[46960100]    <1>       inc   byte [createfile_wfc] ; 1
4514 0000CC35 EB24              <1>       jmp   short loc_createfile_update_fat_cluster_n2
4515                            <1>
4516                            <1> loc_createfile_dsk_wr_err:
4517                            <1>       ; 16/10/2016 (1Dh -> 18)
4518                            <1>       ; 23/03/2016
4519 0000CC37 B812000000        <1>       mov   eax, 18 ; Drive not ready or write error !
4520 0000CC3C E9BD000000        <1>       jmp   loc_createfile_stc_retn
4521                            <1>
4522                            <1> loc_createfile_update_fat_cluster_n1:
4523 0000CC41 A1[38960100]      <1>       mov   eax, [createfile_PCluster]
4524 0000CC46 8B0D[34960100]    <1>       mov   ecx, [createfile_Cluster]
4525 0000CC4C E84F040000        <1>       call  update_cluster
4526 0000CC51 7308              <1>       jnc   short loc_createfile_update_fat_cluster_n2
4527 0000CC53 09C0              <1>       or    eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4528 0000CC55 0F85A3000000      <1>       jnz   loc_createfile_stc_retn
4529                            <1>
4530                            <1> loc_createfile_update_fat_cluster_n2:
4531 0000CC5B A1[34960100]      <1>         mov eax, [createfile_Cluster]
4532 0000CC60 B9FFFFFF0F        <1>       mov   ecx, 0FFFFFFFh
4533 0000CC65 E836040000        <1>       call  update_cluster
```

```
4534 0000CC6A 734E                <1>       jnc    short loc_createfile_save_fat_buffer_3
4535 0000CC6C 09C0                <1>       or     eax, eax ; EAX = 0 -> cluster value is 0 or eocc
4536 0000CC6E 744A                <1>       jz     short loc_createfile_save_fat_buffer_3
4537                              <1>
4538                              <1> loc_createfile_upd_fat_fcluster_stc_retn:
4539 0000CC70 E989000000          <1>       jmp    loc_createfile_stc_retn
4540                              <1>
4541                              <1> loc_createfile_get_set_wfc_cont:
4542                              <1>       ;movzx ecx, word [esi+LD_BPB+BytesPerSec] ; 512
4543 0000CC75 0FB70D[44960100]    <1>       movzx  ecx, word [createfile_BytesPerSec] ; 512
4544 0000CC7C 01C8                <1>       add    eax, ecx
4545 0000CC7E 48                  <1>       dec    eax  ; add eax, 511
4546 0000CC7F 29D2                <1>       sub    edx, edx
4547 0000CC81 F7F1                <1>       div    ecx
4548 0000CC83 0FB61D[3D960100]    <1>       movzx  ebx, byte [createfile_SecPerClust]
4549 0000CC8A 01D8                <1>       add    eax, ebx
4550 0000CC8C 48                  <1>       dec    eax  ; add eax, SecPerClust - 1
4551 0000CC8D 6631D2              <1>       xor    dx, dx
4552 0000CC90 F7F3                <1>       div    ebx
4553                              <1>
4554                              <1> loc_createfile_set_cluster_count:
4555 0000CC92 A3[40960100]        <1>       mov    [createfile_CCount], eax
4556                              <1>
4557 0000CC97 BF00000700          <1>       mov    edi, Cluster_Buffer
4558 0000CC9C 89C8                <1>       mov    eax, ecx ; Bytes per Sector
4559 0000CC9E F7E3                <1>       mul    ebx ; Sectors per Cluster
4560                              <1>       ; EAX = Bytes per Cluster
4561 0000CCA0 89C1                <1>       mov    ecx, eax
4562 0000CCA2 C1E902              <1>       shr    ecx, 2 ; dword count
4563 0000CCA5 31C0                <1>       xor    eax, eax
4564 0000CCA7 F3AB                <1>       rep    stosd ; clear cluster buffer
4565                              <1>
4566 0000CCA9 A1[34960100]        <1>       mov    eax, [createfile_Cluster] ; 24/03/2016
4567                              <1>
4568 0000CCAE 89D9                <1>       mov    ecx, ebx
4569                              <1>
4570                              <1> loc_createfile_get_set_wf_fclust_cont:
4571 0000CCB0 83E802              <1>       sub    eax, 2
4572 0000CCB3 F7E1                <1>       mul    ecx
4573                              <1>       ; EAX = Logical DOS disk address (offset)
4574 0000CCB5 E95DFFFFFF          <1>         jmp     loc_createfile_write_fcluster_to_disk
4575                              <1>
4576                              <1> loc_createfile_save_fat_buffer_3:
4577                              <1>       ; byte [FAT_BuffValidData] = 2
4578 0000CCBA E89E060000          <1>       call   save_fat_buffer
4579 0000CCBF 723D                <1>       jc     loc_createfile_stc_retn
4580                              <1>
4581                              <1>       ; 21/03/2016
4582 0000CCC1 803D[FA910100]01    <1>       cmp    byte [FAT_ClusterCounter], 1
4583 0000CCC8 721B                <1>       jb     short loc_createfile_save_fat_buffer_4
4584                              <1>
4585                              <1>       ; ESI = Logical DOS Drive Description Table address
4586 0000CCCA A1[FA910100]        <1>       mov    eax, [FAT_ClusterCounter]
4587 0000CCCF 66BB01FF            <1>       mov    bx, 0FF01h ; add free clusters
4588 0000CCD3 E81A070000          <1>       call   calculate_fat_freespace
4589                              <1>
4590                              <1>       ;inc   eax ; 0FFFFFFFFh -> 0 ; recalculation is needed!
4591                              <1>       ;jnz   short loc_createfile_save_fat_buffer_4
4592                              <1>
4593                              <1>       ; ecx > 0 -> Recalculation is needed
4594 0000CCD8 09C9                <1>       or     ecx, ecx
4595 0000CCDA 7409                <1>       jz     short loc_createfile_save_fat_buffer_4
4596                              <1>
4597 0000CCDC 66BB00FF            <1>       mov    bx, 0FF00h ; ; recalculate free space
4598 0000CCE0 E80D070000          <1>       call   calculate_fat_freespace
4599                              <1>
4600                              <1> loc_createfile_save_fat_buffer_4:
4601 0000CCE5 FF0D[40960100]      <1>       dec    dword [createfile_CCount]
4602                              <1>       ;jz    short loc_createfile_upd_dir_modif_date_time
4603 0000CCEB 743F                <1>       jz     short loc_createfile_stc_retn_cc ; 31/03/2016
4604                              <1>
4605                              <1> loc_createfile_get_set_write_next_cluster:
4606 0000CCED E8DB020000          <1>       call   get_first_free_cluster
4607 0000CCF2 720A                <1>       jc     short loc_createfile_stc_retn
4608                              <1>
4609                              <1> loc_createfile_get_set_write_next_cluster_1:
4610 0000CCF4 83F8FF              <1>       cmp    eax, 0FFFFFFFFh
4611 0000CCF7 7213                <1>       jb     short loc_createfile_get_set_write_next_cluster_2
4612                              <1>
4613                              <1> loc_createfile_wnc_insufficient_disk_space:
4614 0000CCF9 B827000000          <1>       mov    eax, 27h ; Insufficient disk space
4615                              <1>
4616                              <1> loc_createfile_stc_retn:
4617 0000CCFE 803D[46960100]01    <1>       cmp    byte [createfile_wfc], 1
4618 0000CD05 7324                <1>       jnb    short loc_createfile_err_retn
4619 0000CD07 C3                  <1>       retn
4620                              <1>
4621                              <1> loc_createfile_wnc_inv_format_retn:
4622                              <1>       ;mov   eax, 28
4623 0000CD08 B01C                <1>       mov    al, 28 ; Invalid format
4624 0000CD0A EBF2                <1>       jmp    short loc_createfile_stc_retn
4625                              <1>
4626                              <1> loc_createfile_get_set_write_next_cluster_2:
4627 0000CD0C 83F802              <1>       cmp    eax, 2
4628 0000CD0F 72F7                <1>       jb     short loc_createfile_wnc_inv_format_retn
4629                              <1>
4630                              <1> loc_createfile_get_set_write_next_cluster_3:
4631 0000CD11 8B0D[34960100]      <1>       mov    ecx, [createfile_Cluster]
4632 0000CD17 A3[34960100]        <1>       mov    [createfile_Cluster], eax
4633 0000CD1C 890D[38960100]      <1>       mov    [createfile_PCluster], ecx
4634 0000CD22 0FB60D[3D960100]    <1>       movzx  ecx, byte [createfile_SecPerClust]
4635 0000CD29 EB85                <1>       jmp    short loc_createfile_get_set_wf_fclust_cont
4636                              <1>
4637                              <1> loc_createfile_err_retn:
4638 0000CD2B F9                  <1>       stc
```

```
4639                                    <1>
4640                                    <1> ;loc_createfile_upd_dir_modif_date_time:
4641                                    <1> loc_createfile_stc_retn_cc: ; 31/03/2016
4642 0000CD2C 9C                       <1>       pushf  ; cpu is here for an error return or completion
4643 0000CD2D 50                       <1>       push   eax ; error code if cf = 1
4644                                    <1>
4645                                    <1>       ;call  update_parent_dir_lmdt
4646                                    <1>
4647                                    <1> ;loc_createfile_stc_retn_cc:
4648 0000CD2E A1[FA910100]             <1>       mov    eax, [FAT_ClusterCounter]
4649 0000CD33 09C0                     <1>       or     eax, eax
4650 0000CD35 741A                     <1>       jz     short loc_createfile_stc_retn_pop_eax
4651 0000CD37 8A3D[DE8A0100]           <1>       mov    bh, [Current_Drv]
4652 0000CD3D B301                     <1>       mov    bl, 01h ; BL = 1 -> add clusters
4653                                    <1>       ; NOTE: EAX value will be added to Free Cluster Count
4654                                    <1>       ; (If EAX value is negative, Free Cluster Count will be decreased)
4655 0000CD3F E8AE060000               <1>       call   calculate_fat_freespace
4656                                    <1>         ; ESI = Logical DOS Drive Description Table Address
4657                                    <1>         ;jc  short loc_createfile_stc_retn_pop_eax_cf
4658 0000CD44 21C9                     <1>       and    ecx, ecx ; cx = 0 -> valid free sector count
4659 0000CD46 7409                     <1>       jz     short loc_createfile_stc_retn_pop_eax
4660                                    <1>
4661                                    <1> loc_createfile_stc_retn_recalc_FAT_freespace:
4662 0000CD48 66BB00FF                 <1>       mov    bx, 0FF00h ; bh = 0FFh ->
4663                                    <1>       ; ESI = Logical DOS Drv DT Addr
4664                                    <1>       ; BL = 0 -> Recalculate
4665 0000CD4C E8A1060000               <1>       call   calculate_fat_freespace
4666                                    <1>
4667                                    <1> loc_createfile_stc_retn_pop_eax:
4668 0000CD51 58                       <1>       pop    eax
4669 0000CD52 9D                       <1>       popf
4670 0000CD53 7218                     <1>       jc     short loc_createfile_retn
4671                                    <1>
4672                                    <1> loc_createfile_retn_fcluster:
4673 0000CD55 A1[2C960100]             <1>       mov    eax, [createfile_FFCluster]
4674 0000CD5A BB[28960100]             <1>       mov    ebx, createfile_size
4675                                    <1>       ;movzx ecx, byte [esi+LD_BPB+SecPerClust]
4676 0000CD5F 0FB60D[3D960100]         <1>       movzx  ecx, byte [createfile_SecPerClust] ; 23/03/2016
4677 0000CD66 0FB715[3E960100]         <1>       movzx  edx, word [createfile_DirIndex]
4678                                    <1>
4679                                    <1> loc_createfile_retn:
4680 0000CD6D C3                       <1>       retn
4681                                    <1>
4682                                    <1> create_fs_file:
4683                                    <1>       ; temporary (21/03/2016)
4684 0000CD6E C3                       <1>       retn
4685                                    <1>
4686                                    <1> delete_fs_file:
4687                                    <1>       ; temporary (28/02/2016)
4688 0000CD6F C3                       <1>       retn
4689                                    <1>
4690                                    <1> rename_fs_file_or_directory:
4691 0000CD70 C3                       <1>       retn
4692                                    <1>
4693                                    <1> make_fs_directory:
4694                                    <1>       ; temporary (21/02/2016)
4695 0000CD71 C3                       <1>       retn
4696                                    <1>
4697                                    <1> add_new_fs_section:
4698                                    <1>       ; temporary (11/03/2016)
4699 0000CD72 C3                       <1>       retn
4700                                    <1>
4701                                    <1> delete_fs_directory_entry:
4702                                    <1>       ; temporary (11/03/2016)
4703 0000CD73 C3                       <1>       retn
4704                                    <1>
4705                                    <1> csftdf2_read_fs_file_sectors:
4706                                    <1>       ; temporary (19/03/2016)
4707 0000CD74 C3                       <1>       retn
4708                                    <1>
4709                                    <1> csftdf2_write_fs_file_sectors:
4710                                    <1>       ; temporary (19/03/2016)
4711 0000CD75 C3                       <1>       retn
3091                                        %include 'trdosk5.s' ; 24/01/2016
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - File System Procedures : trdosk5s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 23/10/2016
   5                                    <1> ; ----------------------------------------------------------------------------
   6                                    <1> ; Beginning: 24/01/2016
   7                                    <1> ; ----------------------------------------------------------------------------
   8                                    <1> ; Assembler: NASM version 2.11 (trdos386.s)
   9                                    <1> ; ----------------------------------------------------------------------------
  10                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
  11                                    <1> ; DRV_FAT.ASM (21/08/2011)
  12                                    <1> ; ****************************************************************************
  13                                    <1> ; DRV_FAT.ASM (c) 2005-2011 Erdogan TAN [ 07/07/2009 ] Last Update: 21/08/2011
  14                                    <1>
  15                                    <1> get_next_cluster:
  16                                    <1>       ; 15/10/2016
  17                                    <1>       ; 23/03/2016
  18                                    <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
  19                                    <1>       ; 05/07/2011
  20                                    <1>       ; 07/07/2009
  21                                    <1>       ; 2005
  22                                    <1>       ; INPUT ->
  23                                    <1>       ;     EAX = Cluster Number (32 bit)
  24                                    <1>       ;     ESI = Logical DOS Drive Parameters Table
  25                                    <1>       ; OUTPUT ->
  26                                    <1>       ;     cf = 0 -> No Error, EAX valid
  27                                    <1>       ;     cf = 1 & EAX = 0 -> End Of Cluster Chain
  28                                    <1>       ;     cf = 1 & EAX > 0 -> Error
  29                                    <1>       ;     ECX = Current/Previous cluster (if CF = 0)
  30                                    <1>       ;     EAX = Next Cluster Number (32 bit)
  31                                    <1>       ;
```

```
                        <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                        <1>
0000CD76 A3[EE910100]   <1>       mov    [FAT_CurrentCluster], eax
                        <1> check_next_cluster_fat_type:
0000CD7B 29D2           <1>       sub    edx, edx ; 0
0000CD7D 807E0302       <1>         cmp     byte [esi+LD_FATType], 2
0000CD81 7250           <1>       jb     short get_FAT12_next_cluster
0000CD83 0F87AF000000   <1>         ja      get_FAT32_next_cluster
                        <1> get_FAT16_next_cluster:
0000CD89 BB00030000     <1>       mov    ebx, 300h ;768
0000CD8E F7F3           <1>       div    ebx
                        <1>       ; EAX = Count of 3 FAT sectors
                        <1>       ; EDX = Cluster Offset (< 768)
0000CD90 66D1E2         <1>       shl    dx, 1 ; Multiply by 2
0000CD93 89D3           <1>       mov    ebx, edx ; Byte Offset
0000CD95 81C3001C0900   <1>       add    ebx, FAT_Buffer
0000CD9B 66BA0300       <1>       mov    dx, 3
0000CD9F F7E2           <1>       mul    edx
                        <1>       ; EAX = FAT Sector (<= 256)
                        <1>       ; EDX = 0
0000CDA1 8A0E           <1>       mov    cl, [esi+LD_Name]
0000CDA3 803D[F2910100]00 <1>     cmp    byte [FAT_BuffValidData], 0
0000CDAA 0F86CC000000   <1>         jna     load_FAT_sectors0
0000CDB0 3A0D[F3910100] <1>       cmp    cl, [FAT_BuffDrvName]
0000CDB6 0F85C0000000   <1>         jne     load_FAT_sectors0
0000CDBC 3B05[F6910100] <1>       cmp    eax, [FAT_BuffSector]
0000CDC2 0F85BA000000   <1>         jne     load_FAT_sectors1
                        <1>       ;movzx eax, word [ebx]
0000CDC8 668B03         <1>       mov    ax, [ebx]
                        <1>       ; 01/02/2016
                        <1>       ; DRV_FAT.ASM (21/08/2011) had a FATal bug here !
                        <1>       ; (cmp ah, 0Fh) ! (ax >= FF7h)
                        <1>       ; (how can i do a such mistake!?)
                        <1>       ;cmp   al, 0F7h
                        <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
                        <1>       ;cmp   ah, 0FFh
                        <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
0000CDCB 6683F8F7       <1>       cmp    ax, 0FFF7h
0000CDCF 725A           <1>       jb     short loc_pass_gnc_FAT16_eoc_check
                        <1>       ; ax >= FFF7h (cluster 0002h to FFF6h is valid, in use)
0000CDD1 EB56           <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
                        <1>
                        <1> get_FAT12_next_cluster:
0000CDD3 BB00040000     <1>       mov    ebx, 400h ;1024
0000CDD8 F7F3           <1>       div    ebx
                        <1>       ; EAX = Count of 3 FAT sectors
                        <1>       ; EDX = Cluster Offset (< 1024)
0000CDDA 6650           <1>       push   ax
0000CDDC 66B80300       <1>       mov    ax, 3
0000CDE0 66F7E2         <1>       mul    dx     ; Multiply by 3
0000CDE3 66D1E8         <1>       shr    ax, 1  ; Divide by 2
0000CDE6 6689C3         <1>         mov  bx, ax       ; Byte Offset
0000CDE9 81C3001C0900   <1>       add    ebx, FAT_Buffer
0000CDEF 6658           <1>       pop    ax
0000CDF1 66BA0300       <1>       mov    dx, 3
0000CDF5 F7E2           <1>       mul    edx
                        <1>       ; EAX = FAT Sector (<= 12)
                        <1>       ; EDX = 0
0000CDF7 8A0E           <1>       mov    cl, [esi+LD_Name]
0000CDF9 803D[F2910100]00 <1>     cmp    byte [FAT_BuffValidData], 0
0000CE00 767A           <1>       jna    short load_FAT_sectors0
0000CE02 3A0D[F3910100] <1>       cmp    cl, [FAT_BuffDrvName]
0000CE08 7572           <1>       jne    short load_FAT_sectors0
0000CE0A 3B05[F6910100] <1>       cmp    eax, [FAT_BuffSector]
0000CE10 7570           <1>       jne    short load_FAT_sectors1
0000CE12 A1[EE910100]   <1>       mov    eax, [FAT_CurrentCluster]
0000CE17 66D1E8         <1>       shr    ax, 1
                        <1>       ;movzx eax, word [ebx]
0000CE1A 668B03         <1>       mov    ax, [ebx]
0000CE1D 7314           <1>       jnc    short get_FAT12_nc_even
0000CE1F 66C1E804       <1>       shr    ax, 4
                        <1> loc_gnc_fat12_eoc_check:
                        <1>       ;cmp   al, 0F7h
                        <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
                        <1>       ;cmp   ah, 0Fh
                        <1>       ;jb    short loc_pass_gnc_FAT16_eoc_check
0000CE23 663DF70F       <1>       cmp    ax, 0FF7h
0000CE27 7202           <1>       jb     short loc_pass_gnc_FAT16_eoc_check
                        <1>       ; ax >= FF7h (cluster 0002h to FF6h is valid, in use)
                        <1>
                        <1> loc_pass_gnc_FAT16_eoc_check_xor_eax:
0000CE29 31C0           <1>       xor    eax, eax ; 0
                        <1> loc_pass_gnc_FAT16_eoc_check:
                        <1> loc_pass_gnc_FAT32_eoc_check:
0000CE2B 8B0D[EE910100] <1>       mov    ecx, [FAT_CurrentCluster]
0000CE31 F5             <1>       cmc
0000CE32 C3             <1>       retn
                        <1>
                        <1> get_FAT12_nc_even:
0000CE33 80E40F         <1>       and    ah, 0Fh
0000CE36 EBEB           <1>       jmp    short loc_gnc_fat12_eoc_check
                        <1>
                        <1> get_FAT32_next_cluster:
0000CE38 BB80010000     <1>       mov    ebx, 180h ;384
0000CE3D F7F3           <1>       div    ebx
                        <1>       ; EAX = Count of 3 FAT sectors
                        <1>       ; EDX = Cluster Offset (< 384)
0000CE3F 66C1E202       <1>       shl    dx, 2  ; Multiply by 4
0000CE43 89D3           <1>       mov    ebx, edx ; Byte Offset
0000CE45 81C3001C0900   <1>       add    ebx, FAT_Buffer
0000CE4B 66BA0300       <1>       mov    dx, 3
0000CE4F F7E2           <1>       mul    edx
                        <1>         ; EAX = FAT Sector (<= 2097152) ; (FFFFFF7h * 4) / 512
                        <1>       ;      for 32KB cluster size:
                        <1>       ;      EAX <= 1024 = (4GB / 32KB) * 4) / 512
```

```
                                <1>       ; EDX = 0
0000CE51 8A0E                   <1>       mov    cl, [esi+LD_Name]
0000CE53 803D[F2910100]00       <1>       cmp    byte [FAT_BuffValidData], 0
0000CE5A 7620                   <1>       jna    short load_FAT_sectors0
0000CE5C 3A0D[F3910100]         <1>       cmp    cl, [FAT_BuffDrvName]
0000CE62 7518                   <1>       jne    short load_FAT_sectors0
0000CE64 3B05[F6910100]         <1>       cmp    eax, [FAT_BuffSector] ; 0, 3, 6, 9 ...
0000CE6A 7516                   <1>       jne    short load_FAT_sectors1
0000CE6C 8B03                   <1>       mov    eax, [ebx]
0000CE6E 25FFFFFF0F             <1>       and    eax, 0FFFFFFFh ; 28 bit Cluster
0000CE73 3DF7FFFF0F             <1>       cmp    eax, 0FFFFFF7h
0000CE78 72B1                   <1>       jb     short loc_pass_gnc_FAT32_eoc_check
                                <1>       ; eax >= FFFFFF7h (cluster 0002h to FFFFFF6h is valid)
0000CE7A EBAD                   <1>       jmp    short loc_pass_gnc_FAT16_eoc_check_xor_eax
                                <1>
                                <1> load_FAT_sectors0:
0000CE7C 880D[F3910100]         <1>       mov    [FAT_BuffDrvName], cl
                                <1> load_FAT_sectors1:
0000CE82 A3[F6910100]           <1>       mov    [FAT_BuffSector], eax
0000CE87 89C3                   <1>       mov    ebx, eax
0000CE89 034660                 <1>         add     eax, [esi+LD_FATBegin]
0000CE8C 807E0302               <1>       cmp    byte [esi+LD_FATType], 2
0000CE90 7706                   <1>         ja      short load_FAT_sectors3
0000CE92 0FB74E1C               <1>       movzx  ecx, word [esi+LD_BPB+BPB_FATSz16]
0000CE96 EB03                   <1>       jmp    short load_FAT_sectors4
                                <1> load_FAT_sectors3:
0000CE98 8B4E2A                 <1>       mov    ecx, [esi+LD_BPB+BPB_FATSz32]
                                <1> load_FAT_sectors4:
0000CE9B 29D9                   <1>       sub    ecx, ebx ; [FAT_BuffSector]
0000CE9D 83F903                 <1>         cmp     ecx, 3
0000CEA0 7605                   <1>         jna     short load_FAT_sectors5
0000CEA2 B903000000             <1>       mov    ecx, 3
                                <1> load_FAT_sectors5:
0000CEA7 BB001C0900             <1>       mov    ebx, FAT_Buffer
0000CEAC E8EF5C0000             <1>       call   disk_read
0000CEB1 730D                   <1>       jnc    short load_FAT_sectors_ok
                                <1>       ; 15/10/2016 (15h -> 17)
                                <1>       ; 23/03/2016 (15h)
0000CEB3 B811000000             <1>       mov    eax, 17 ; Drive not ready or read error
0000CEB8 C605[F2910100]00       <1>       mov    byte [FAT_BuffValidData], 0
0000CEBF C3                     <1>       retn
                                <1> load_FAT_sectors_ok:
0000CEC0 C605[F2910100]01       <1>       mov    byte [FAT_BuffValidData], 1
0000CEC7 A1[EE910100]           <1>       mov    eax, [FAT_CurrentCluster]
0000CECC E9AAFEFFFF             <1>         jmp     check_next_cluster_fat_type
                                <1>
                                <1> load_FAT_root_directory:
                                <1>       ; 23/10/2016
                                <1>       ; 15/10/2016
                                <1>       ; 07/02/2016
                                <1>       ; 02/02/2016
                                <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 21/05/2011
                                <1>       ; 22/08/2009
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     ESI = Logical DOS Drive Description Table
                                <1>       ; OUTPUT ->
                                <1>       ;     cf = 1 -> Root directory could not be loaded
                                <1>       ;         EAX > 0 -> Error number
                                <1>       ;     cf = 0 -> EAX = 0
                                <1>       ;     ECX = Directory buffer size in sectors (CL)
                                <1>       ;     EBX = Directory buffer address
                                <1>       ;     NOTE: DirBuffer_Size is in bytes ! (word)
                                <1>       ;
                                <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                                <1>
                                <1>       ; NOTE: Only for FAT12 and FAT16 file systems !
                                <1>       ; (FAT32 fs root dir must be loaded as sub directory)
                                <1>
0000CED1 8A1E                   <1>       mov    bl, [esi+LD_Name]
0000CED3 8A7E03                 <1>       mov    bh, [esi+LD_FATType]
                                <1>
                                <1>       ;mov   [DirBuff_DRV], bl
                                <1>       ;mov   [DirBuff_FATType], bh
0000CED6 66891D[02920100]       <1>       mov    [DirBuff_DRV], bx
                                <1>
                                <1>       ;cmp   bh, 2
                                <1>       ;ja    short load_FAT32_root_dir0 ; FAT32 root dir
                                <1>
                                <1> load_FAT_root_dir0: ; 23/10/2016
0000CEDD 0FB75617               <1>       movzx  edx, word [esi+LD_BPB+RootDirEnts]
                                <1>
                                <1>       ;or    dx, dx ; 0 for FAT32 file systems
                                <1>       ;jz    short load_FAT32_root_dir0 ; FAT32 root dir
                                <1>
0000CEE1 6681FA0002             <1>       cmp    dx, 512 ; Number of Root Dir Entries
0000CEE6 7414                   <1>       je     short lrd_mov_ecx_32
0000CEE8 89D0                   <1>       mov    eax, edx
                                <1>       ; 23/10/2016
0000CEEA 89C1                   <1>       mov    ecx, eax
0000CEEC 6683C10F               <1>       add    cx, 15 ; round up
0000CEF0 66C1E904               <1>       shr    cx, 4  ; 16 entries per sector (512/32)
                                <1>       ; ecx = Root directory size in sectors
0000CEF4 66C1E005               <1>       shl    ax, 5 ; Root directory size in bytes
0000CEF8 664A                   <1>       dec    dx    ; Last entry number of root dir
                                <1>       ; cx = Dir Buffer sector count
0000CEFA EB0B                   <1>       jmp    short lrd_check_dir_buffer
                                <1>
                                <1> lrd_mov_ecx_32:
0000CEFC B920000000             <1>       mov    ecx, 32
0000CF01 664A                   <1>       dec    dx ; 511
0000CF03 66B80040               <1>       mov    ax, 32*512
                                <1>
                                <1> lrd_check_dir_buffer:
```

```
0000CF07 29DB                <1>       sub    ebx, ebx ; 0
0000CF09 881D[04920100]      <1>       mov    [DirBuff_ValidData], bl ; 0
0000CF0F 668915[07920100]    <1>       mov    [DirBuff_LastEntry], dx
0000CF16 891D[09920100]      <1>       mov    [DirBuff_Cluster], ebx ; 0
0000CF1C 66A3[0D920100]      <1>       mov    [DirBuffer_Size], ax
                             <1>
0000CF22 8B4664              <1>       mov    eax, [esi+LD_ROOTBegin]
                             <1> read_directory:
0000CF25 BB00000800          <1>       mov    ebx, Directory_Buffer
0000CF2A 51                  <1>       push   ecx ; Directory buffer sector count
0000CF2B 53                  <1>       push   ebx
0000CF2C E86F5C0000          <1>       call   disk_read
0000CF31 5B                  <1>       pop    ebx
0000CF32 720B                <1>       jc     short load_DirBuff_error
                             <1>
                             <1> validate_DirBuff_and_return:
0000CF34 59                  <1>       pop    ecx ; Number of loaded sectors
0000CF35 C605[04920100]01    <1>       mov    byte [DirBuff_ValidData], 1
0000CF3C 31C0                <1>       xor    eax, eax ; 0 = no error
0000CF3E C3                  <1>       retn
                             <1>
                             <1> load_DirBuff_error:
0000CF3F 89C8                <1>       mov    eax, ecx ; remaining sectors
0000CF41 59                  <1>       pop    ecx ; sector count
0000CF42 29C1                <1>       sub    ecx, eax ; Number of loaded sectors
                             <1>       ; 15/10/2016 (15h -> 17)
0000CF44 B811000000          <1>       mov    eax, 17 ; DRV NOT READY OR READ ERROR !
0000CF49 F9                  <1>       stc
0000CF4A C3                  <1>         retn
                             <1>
                             <1> load_FAT32_root_directory:
                             <1>       ; 02/02/2016 (TRDOS 386 =  TRDOS v2.0)
                             <1>       ;
                             <1>       ; INPUT ->
                             <1>       ;      ESI = Logical DOS Drive Description Table
                             <1>       ; OUTPUT ->
                             <1>       ;      cf = 1 -> Root directory could not be loaded
                             <1>       ;           EAX > 0 -> Error number
                             <1>       ;      cf = 0 -> EAX = 0
                             <1>       ;      ECX = Directory buffer size in sectors (CL)
                             <1>       ;      EBX = Directory buffer address
                             <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
                             <1>       ;
                             <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                             <1>
                             <1>
0000CF4B 8A1E                <1>       mov    bl, [esi+LD_Name]
0000CF4D 8A7E03              <1>       mov    bh, [esi+LD_FATType]
                             <1>
                             <1>       ;mov   [DirBuff_DRV], bl
                             <1>       ;mov   [DirBuff_FATType], bh
0000CF50 66891D[02920100]    <1>       mov    [DirBuff_DRV], bx
                             <1>
                             <1> load_FAT32_root_dir0:
0000CF57 8B4632              <1>       mov    eax, [esi+LD_BPB+FAT32_RootFClust]
0000CF5A EB0C                <1>       jmp    short load_FAT_sub_dir0
                             <1>
                             <1> load_FAT_sub_directory:
                             <1>       ; 01/02/2016 (TRDOS 386 =  TRDOS v2.0)
                             <1>       ; 05/07/2011
                             <1>       ; 23/08/2009
                             <1>       ;
                             <1>       ; INPUT ->
                             <1>       ;      ESI = Logical DOS Drive Description Table
                             <1>       ;      EAX = Cluster Number
                             <1>       ; OUTPUT ->
                             <1>       ;      cf = 1 -> Sub directory could not be loaded
                             <1>       ;           EAX > 0 -> Error number
                             <1>       ;      cf = 0 -> EAX = 0
                             <1>       ;      ECX = Directory buffer size in sectors (CL)
                             <1>       ;      EBX = Directory buffer address
                             <1>       ;
                             <1>       ;      NOTE: DirBuffer_Size is in bytes ! (word)
                             <1>       ;
                             <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
                             <1>
0000CF5C 8A1E                <1>       mov    bl, [esi+LD_Name]
0000CF5E 8A7E03              <1>       mov    bh, [esi+LD_FATType]
                             <1>
                             <1>       ;mov   [DirBuff_DRV], bl
                             <1>       ;mov   [DirBuff_FATType], bh
0000CF61 66891D[02920100]    <1>       mov    [DirBuff_DRV], bx
                             <1>
                             <1> load_FAT_sub_dir0:
0000CF68 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
                             <1>
0000CF6C 882D[04920100]      <1>       mov    [DirBuff_ValidData], ch ; 0
0000CF72 A3[09920100]        <1>       mov    [DirBuff_Cluster], eax
                             <1>
0000CF77 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
0000CF7B F7E1                <1>       mul    ecx
0000CF7D C1E805              <1>       shr    eax, 5 ; directory entry count (dir size / 32)
0000CF80 6648                <1>       dec    ax ; last entry
0000CF82 66A3[07920100]      <1>       mov    [DirBuff_LastEntry], ax
                             <1>
0000CF88 A1[09920100]        <1>       mov    eax, [DirBuff_Cluster]
0000CF8D 83E802              <1>       sub    eax, 2
0000CF90 F7E1                <1>       mul    ecx
0000CF92 034668              <1>       add    eax, [esi+LD_DATABegin]
                             <1>       ; ecx = sector per cluster (dir buffer size = 32 sectors)
0000CF95 EB8E                <1>       jmp    short read_directory
                             <1>
                             <1> ; DRV_FS.ASM
                             <1>
                             <1> load_current_FS_directory:
```

```
0000CF97 C3                <1>       retn
                           <1> load_FS_root_directory:
0000CF98 C3                <1>       retn
                           <1> load_FS_sub_directory:
0000CF99 C3                <1>       retn
                           <1>
                           <1> read_cluster:
                           <1>       ; 15/10/2016
                           <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>
0000CF9A 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
0000CF9E 807E0300          <1>       cmp    byte [esi+LD_FATType], 0
0000CFA2 761C              <1>       jna    short read_fs_cluster
                           <1>
                           <1> read_fat_file_sectors: ; 18/03/2016
0000CFA4 83E802            <1>       sub    eax, 2 ; Beginning cluster number is always 2
0000CFA7 0FB65613          <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
0000CFAB F7E2              <1>       mul    edx
0000CFAD 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>
0000CFB0 E8EB5B0000        <1>       call   disk_read
0000CFB5 7306              <1>       jnc    short rclust_retn
                           <1>
                           <1>       ; 15/10/2016 (15h -> 17)
0000CFB7 B811000000        <1>       mov    eax, 17 ; Drive not ready or read error !
0000CFBC C3                <1>       retn
                           <1>
                           <1> rclust_retn:
0000CFBD 29C0              <1>       sub    eax, eax ; 0
0000CFBF 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)
0000CFC0 B980000000        <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
0000CFC5 E801000000        <1>       call   read_fs_sectors
0000CFCA C3                <1>       retn
                           <1>
                           <1> read_fs_sectors:
                           <1>       ; 15/02/2016 (TRDOS 386 =  TRDOS v2.0)
0000CFCB F9                <1>       stc
0000CFCC 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>
0000CFCD 8B4678            <1>       mov    eax, [esi+LD_Clusters]
0000CFD0 40                <1>       inc    eax ; add eax, 1
0000CFD1 A3[8C940100]      <1>       mov    [gffc_last_free_cluster], eax
                           <1>
0000CFD6 31DB              <1>       xor    ebx, ebx ; 0 ; 02/03/2016
                           <1>
0000CFD8 807E0302          <1>       cmp    byte [esi+LD_FATType], 2
0000CFDC 760E              <1>       jna    short loc_gffc_get_first_fat_free_cluster0
                           <1>
                           <1> loc_gffc_get_first_fat32_free_cluster:
                           <1>       ; 02/03/2016
0000CFDE E844060000        <1>       call   get_fat32_fsinfo_sector_parms
0000CFE3 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
                        <1>       ;(from FAT32 FSInfo sector)
0000CFE5 89D0           <1>       mov    eax, edx ; FSI_Next_Free (First Free Cluster)
0000CFE7 83F8FF         <1>       cmp    eax, 0FFFFFFFFh ; invalid (unknown) !
0000CFEA 7205           <1>       jb     short loc_gffc_get_first_fat_free_cluster1
                        <1>
                        <1>       ; Start from the 1st cluster of the FAT(32) file system
                        <1> loc_gffc_get_first_fat_free_cluster0:
0000CFEC B802000000     <1>       mov    eax, 2
                        <1>       ;xor   edx, edx
                        <1>
                        <1> loc_gffc_get_first_fat_free_cluster1:
0000CFF1 53             <1>       push   ebx ; 02/03/2016
                        <1>
                        <1> loc_gffc_get_first_fat_free_cluster2:
0000CFF2 A3[88940100]   <1>       mov    [gffc_first_free_cluster], eax
0000CFF7 A3[84940100]   <1>       mov    [gffc_next_free_cluster], eax
                        <1>
                        <1>       ; EBX = FAT32 FSINFO sector buffer address
                        <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
                        <1>       ; FAT32 FSINFO sector buffer is invalid.)
                        <1>
                        <1> loc_gffc_get_first_fat_free_cluster3:
0000CFFC E875FDFFFF     <1>       call   get_next_cluster
0000D001 7307           <1>       jnc    short loc_gffc_get_first_fat_free_cluster4
0000D003 09C0           <1>       or     eax, eax
0000D005 740B           <1>       jz     short loc_gffc_first_free_fat_cluster_next
0000D007 5B             <1>       pop    ebx ; 02/03/2016
0000D008 F5             <1>       cmc    ; stc
0000D009 C3             <1>       retn
                        <1>
                        <1> loc_gffc_get_first_fat_free_cluster4:
0000D00A 21C0           <1>       and    eax, eax ; next cluster value
0000D00C 7504           <1>       jnz    short loc_gffc_first_free_fat_cluster_next
0000D00E 89C8           <1>       mov    eax, ecx ; current (previous cluster) value
0000D010 EB22           <1>       jmp    short loc_gffc_check_for_set
                        <1>
                        <1> loc_gffc_first_free_fat_cluster_next:
0000D012 A1[84940100]   <1>       mov    eax, [gffc_next_free_cluster]
0000D017 3B05[8C940100] <1>       cmp    eax, [gffc_last_free_cluster]
0000D01D 7308           <1>       jnb    short retn_stc_from_get_first_free_cluster
                        <1> pass_gffc_last_cluster_eax_check:
0000D01F 40             <1>       inc    eax ; add eax, 1
0000D020 A3[84940100]   <1>       mov    [gffc_next_free_cluster], eax
0000D025 EBD5           <1>       jmp    short loc_gffc_get_first_fat_free_cluster3
                        <1>
                        <1> retn_stc_from_get_first_free_cluster:
0000D027 A1[88940100]   <1>       mov    eax, [gffc_first_free_cluster]
0000D02C 83F802         <1>       cmp    eax, 2
0000D02F 7709           <1>       ja     short loc_gffc_check_previous_clusters
0000D031 29C0           <1>       sub    eax, eax
0000D033 48             <1>       dec    eax ; FFFFFFFFh
                        <1>
                        <1> loc_gffc_check_for_set:
                        <1>       ; 02/03/2016
0000D034 5B             <1>       pop    ebx
                        <1>
                        <1>       ; EBX = FAT32 FSINFO sector buffer address
                        <1>       ; (EBX = 0, if the drive has not got FAT32 fs or
                        <1>       ; FAT32 FSINFO sector buffer is invalid.)
                        <1>
0000D035 09DB           <1>       or     ebx, ebx
0000D037 750E           <1>       jnz    short loc_gffc_set_ffree_fat32_cluster
                        <1>
                        <1>       ;cmp   byte [esi+LD_FATType], 3
                        <1>       ;jnb   short loc_gffc_set_ffree_fat32_cluster
                        <1>
                        <1>       ;xor   ebx, ebx ; 0
                        <1>
                        <1> loc_gffc_retn:
0000D039 C3             <1>       retn
                        <1>
                        <1> loc_gffc_check_previous_clusters:
0000D03A 48             <1>       dec    eax ; sub eax, 1
0000D03B A3[8C940100]   <1>       mov    [gffc_last_free_cluster], eax
0000D040 B802000000     <1>       mov    eax, 2
                        <1>       ;xor   edx, edx
0000D045 EBAB           <1>       jmp    short loc_gffc_get_first_fat_free_cluster2
                        <1>
                        <1> loc_gffc_set_ffree_fat32_cluster:
                        <1>       ;call  set_first_free_cluster
                        <1>       ;retn
                        <1>       ;jmp   short set_first_free_cluster
                        <1>
                        <1> set_first_free_cluster:
                        <1>       ; 15/10/2016
                        <1>       ; 23/03/2016
                        <1>       ; 02/03/2016
                        <1>       ; 29/02/2016
                        <1>       ; 26/02/2016
                        <1>       ; 21/02/2016 (TRDOS 386 =  TRDOS v2.0)
                        <1>       ; 21/08/2011 (DRV_FAT.ASM, 'proc_set_first_free_cluster')
                        <1>       ; 11/07/2010
                        <1>       ; INPUT ->
                        <1>       ;      ESI = Logical DOS Drive Description Table address
                        <1>       ;      EAX = First free cluster
                        <1>       ;      EBX = FSINFO sector buffer address
```

```
                                <1>       ;      ;;If EBX > 0, it is FSINFO sector buffer address
                                <1>       ;      ;;EBX = 0, if FSINFO sector is not loaded
                                <1>       ; OUTPUT->
                                <1>       ;      ESI = Logical DOS Drive Description Table address
                                <1>       ;      If EBX > 0, it is FSINFO sector buffer address
                                <1>       ;      EBX = 0, if FSINFO sector could not be loaded
                                <1>       ;      CF = 1 -> Error code in AL (EAX)
                                <1>       ;      CF = 0 -> first free cluster is successfully updated
                                <1>
                                <1>       ;cmp   byte [esi+LD_FATType], 3
                                <1>       ;jb    short loc_sffc_invalid_drive
                                <1>
                                <1>       ; Save First Free Cluster value for 'update_cluster'
0000D047 89463E                 <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; First free Cluster
                                <1>
                                <1>       ;or    ebx, ebx
                                <1>       ;jnz   short loc_sffc_read_fsinfo_sector
                                <1>
0000D04A 813B52526141           <1>       cmp     dword [ebx], 41615252h
0000D050 7540                   <1>       jne    short loc_sffc_read_fsinfo_sector
0000D052 81BBE4010000727241-    <1>       cmp    dword [ebx+484], 61417272h
0000D05B 61                     <1>
0000D05C 7534                   <1>       jne    short loc_sffc_read_fsinfo_sector
                                <1>
0000D05E 3B83EC010000           <1>       cmp    eax, [ebx+492]  ; FSI_Next_Free
0000D064 741F                   <1>       je     short loc_sffc_retn
                                <1>
                                <1> loc_sffc_write_fsinfo_sector:
                                <1>       ; EBX = FSINFO sector buffer
                                <1>       ; [CFS_FAT32FSINFOSEC] is set in 'get_fat32_fsinfo_sector_parms'
0000D066 8983EC010000           <1>       mov    [ebx+492], eax
0000D06C A1[9C940100]           <1>       mov    eax, [CFS_FAT32FSINFOSEC]
0000D071 B901000000             <1>       mov    ecx, 1
0000D076 53                     <1>       push   ebx
0000D077 E8155B0000             <1>       call   disk_write
0000D07C 7208                   <1>       jc      short loc_sffc_read_fsinfo_sector_err1
0000D07E 5B                     <1>       pop    ebx
                                <1>
0000D07F 8B83EC010000           <1>       mov    eax, [ebx+492] ; First (Next) Free Cluster
                                <1>
                                <1> loc_sffc_retn:
0000D085 C3                     <1>       retn
                                <1>
                                <1> ;loc_sffc_invalid_drive:
                                <1> ;     mov    eax, 0Fh ; MSDOS Error : Invalid drive
                                <1> ;     push   edx
                                <1>
                                <1> loc_sffc_read_fsinfo_sector_err1:
0000D086 BB00000000             <1>       mov    ebx, 0
                                <1>       ; 15/10/2016 (1Dh -> 18)
                                <1>       ; 23/03/2016 (1Dh)
0000D08B B812000000             <1>       mov    eax, 18 ; Drive not ready or write error
                                <1>
                                <1> loc_sffc_read_fsinfo_sector_err2:
0000D090 5A                     <1>       pop    edx
0000D091 C3                     <1>       retn
                                <1>
                                <1> loc_sffc_read_fsinfo_sector:
0000D092 50                     <1>       push   eax
                                <1>
0000D093 E88F050000             <1>       call   get_fat32_fsinfo_sector_parms
0000D098 72F6                   <1>       jc     short loc_sffc_read_fsinfo_sector_err2
                                <1>
0000D09A 58                     <1>       pop    eax
                                <1>       ; EDX = First (Next) Free Cluster value from FSINFO sector
                                <1>       ; EAX = First Free Cluster value from 'get_next_cluster'
                                <1>       ; (edx = old value)
0000D09B 39D0                   <1>       cmp    eax, edx ; First free Cluster (eax = new value)
0000D09D 75C7                   <1>       jne    short loc_sffc_write_fsinfo_sector
                                <1>
0000D09F C3                     <1>       retn
                                <1>
                                <1> update_cluster:
                                <1>       ; 23/10/2016
                                <1>       ; 23/03/2016
                                <1>       ; 02/03/2016
                                <1>       ; 01/03/2016
                                <1>       ; 29/02/2016
                                <1>       ; 27/02/2016
                                <1>       ; 26/02/2016
                                <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
                                <1>       ; 11/08/2011
                                <1>       ; 09/02/2005
                                <1>       ; INPUT ->
                                <1>       ;      EAX = Cluster Number
                                <1>       ;      ECX = New Cluster Value
                                <1>       ;      ESI = Logical Dos Drive Parameters Table
                                <1>       ;
                                <1>       ;      /// dword [FAT_ClusterCounter] ///
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;      cf = 0 -> No Error, EAX is valid
                                <1>       ;      cf = 1 & EAX = 0 -> End Of Cluster Chain
                                <1>       ;      cf = 1 & EAX > 0 -> Error
                                <1>       ;             (ECX -> any value)
                                <1>       ;      EAX = Next Cluster
                                <1>       ;      ECX = New Cluster Value
                                <1>       ;
                                <1>       ;      /// [FAT_ClusterCounter] is updated,
                                <1>       ;      /// decreased when a free cluster is assigned,
                                <1>       ;      /// increased if an assigned cluster is freed.
                                <1>       ;
                                <1>       ;
                                <1>       ; (Modified registers: EAX, EBX, -ECX-, EDX)
                                <1>
```

```
0000D0A0 A3[EE910100]        <1>       mov    [FAT_CurrentCluster], eax
0000D0A5 890D[90940100]      <1>       mov    [ClusterValue], ecx
                             <1>
                             <1> loc_update_cluster_check_fat_buffer:
0000D0AB 8A1E                <1>       mov    bl, [esi+LD_Name]
0000D0AD 381D[F3910100]      <1>       cmp    [FAT_BuffDrvName], bl
0000D0B3 741A                <1>       je     short loc_update_cluster_check_fat_type
0000D0B5 803D[F2910100]02    <1>       cmp    byte [FAT_BuffValidData], 2
0000D0BC 0F84C2000000        <1>         je     loc_uc_save_fat_buffer
                             <1>
                             <1> loc_uc_reset_fat_buffer_validation:
0000D0C2 C605[F2910100]00    <1>       mov    byte [FAT_BuffValidData], 0
                             <1>
                             <1> loc_uc_check_fat_type_reset_drvname:
0000D0C9 881D[F3910100]      <1>       mov    [FAT_BuffDrvName], bl
                             <1>
                             <1> loc_update_cluster_check_fat_type:
0000D0CF 29D2                <1>       sub    edx, edx ; 26/02/2016
0000D0D1 8A5E03              <1>       mov    bl, [esi+LD_FATType]
0000D0D4 83F802              <1>       cmp    eax, 2
0000D0D7 0F82BE000000        <1>         jb     update_cluster_inv_data
0000D0DD 80FB02              <1>       cmp    bl, 2
0000D0E0 0F877A010000        <1>         ja     update_fat32_cluster
                             <1>       ;cmp   bl, 1
                             <1>       ;jb    short update_cluster_inv_data
0000D0E6 8B4E78              <1>       mov    ecx, [esi+LD_Clusters]
0000D0E9 41                  <1>       inc    ecx
0000D0EA 890D[FE910100]      <1>       mov    [LastCluster], ecx
0000D0F0 39C8                <1>       cmp    eax, ecx ; dword [LastCluster]
0000D0F2 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:')
0000D0F8 80FB01              <1>       cmp    bl, 1 ; correct comparison is this !
0000D0FB 0F86A2000000        <1>         jna    update_fat12_cluster
                             <1>
                             <1> update_fat16_cluster:
                             <1> pass_uc_fat16_errc:
                             <1>       ;sub   edx, edx
0000D101 BB00030000          <1>       mov    ebx, 300h ;768
0000D106 F7F3                <1>       div    ebx
                             <1>       ; EAX = Count of 3 FAT sectors
                             <1>       ; DX = Cluster offset in FAT buffer
0000D108 6689D3              <1>       mov    bx, dx
0000D10B 66D1E3              <1>       shl    bx, 1 ; Multiply by 2
0000D10E 66BA0300            <1>       mov    dx, 3
0000D112 F7E2                <1>       mul    edx
                             <1>       ; EAX = FAT Sector
                             <1>       ; EDX = 0
                             <1>       ; EBX = Byte offset in FAT buffer
0000D114 8A0D[F2910100]      <1>       mov    cl, [FAT_BuffValidData]
0000D11A 80F902              <1>       cmp    cl, 2
0000D11D 750A                <1>       jne    short loc_uc_check_fat16_buff_sector_load
                             <1>
                             <1> loc_uc_check_fat16_buff_sector_save:
0000D11F 3B05[F6910100]      <1>       cmp    eax, [FAT_BuffSector]
0000D125 755D                <1>       jne    short loc_uc_save_fat_buffer
0000D127 EB15                <1>       jmp    short loc_update_fat16_cell
                             <1>
                             <1> loc_uc_check_fat16_buff_sector_load:
0000D129 80F901              <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
0000D12C 0F85FB010000        <1>         jne    loc_uc_load_fat_sectors
0000D132 3B05[F6910100]      <1>       cmp    eax, [FAT_BuffSector]
0000D138 0F85EF010000        <1>         jne    loc_uc_load_fat_sectors
                             <1>
                             <1> loc_update_fat16_cell:
                             <1> loc_update_fat16_buffer:
0000D13E 81C3001C0900        <1>       add    ebx, FAT_Buffer ; 26/02/2016
                             <1>       ;movzx eax, word [ebx]
0000D144 668B03              <1>       mov    ax, [ebx]
                             <1>       ; 01/03/2016
0000D147 89C2                <1>       mov    edx, eax ; old value of the cluster
0000D149 A3[EE910100]        <1>       mov    [FAT_CurrentCluster], eax
0000D14E 8B0D[90940100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
0000D154 66890B              <1>       mov    [ebx], cx ; 16 bits !
                             <1>
0000D157 C605[F2910100]02    <1>       mov    byte [FAT_BuffValidData], 2
                             <1>
0000D15E 6683F802            <1>       cmp    ax, 2
0000D162 723A                <1>       jb     short return_uc_fat_stc
0000D164 3B05[FE910100]      <1>       cmp    eax, [LastCluster]
0000D16A 7732                <1>       ja     short return_uc_fat_stc
                             <1>
                             <1> loc_fat_buffer_updated:
                             <1>       ; 01/03/2016
0000D16C F8                  <1>       clc
                             <1> loc_fat_buffer_stc_1:
0000D16D 9C                  <1>       pushf
0000D16E 21C9                <1>       and    ecx, ecx
0000D170 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
0000D172 FF05[FA910100]      <1>       inc    dword [FAT_ClusterCounter]
                             <1>
                             <1> loc_fat_buffer_updated_1: ; new value of the cluster > 0
0000D178 09D2                <1>       or     edx, edx ; 02/03/2016
0000D17A 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
```

```
0000D17C FF0D[FA910100]      <1>       dec    dword [FAT_ClusterCounter] ; it may be negative number
                             <1>
                             <1> loc_fat_buffer_updated_2:
0000D182 9D                  <1>       popf
0000D183 C3                  <1>       retn
                             <1>
                             <1> loc_uc_save_fat_buffer:
                             <1>       ; byte [FAT_BuffValidData] = 2
0000D184 E8D4010000          <1>       call   save_fat_buffer
0000D189 0F8297010000        <1>         jc     loc_fat_sectors_rw_error2
                             <1>       ;mov   byte [FAT_BuffValidData], 1
0000D18F A1[EE910100]        <1>       mov    eax, [FAT_CurrentCluster]
                             <1>       ;mov   ecx, [ClusterValue]
                             <1>       ;jmp   short loc_update_cluster_check_fat_buffer
0000D194 8A1E                <1>       mov    bl, [esi+LD_Name] ; 01/03/2016
0000D196 E927FFFFFF          <1>         jmp    loc_uc_reset_fat_buffer_validation
                             <1>
                             <1> update_cluster_inv_data:
                             <1>       ;mov   eax, 0Dh
0000D19B B00D                <1>       mov    al, 0Dh  ; Invalid Data
0000D19D C3                  <1>       retn
                             <1>
                             <1> return_uc_fat_stc:
                             <1>       ; 01/03/2016
0000D19E 31C0                <1>       xor    eax, eax
0000D1A0 F9                  <1>       stc
0000D1A1 EBCA                <1>       jmp    short loc_fat_buffer_stc_1
                             <1>
                             <1> update_fat12_cluster:
                             <1> pass_uc_fat12_errc:
                             <1>       ;sub   edx, edx
0000D1A3 BB00040000          <1>       mov    ebx, 400h ;1024
0000D1A8 F7F3                <1>       div    ebx
                             <1>       ; EAX = Count of 3 FAT sectors
                             <1>       ; DX = Cluster offset in FAT buffer
0000D1AA 66B90300            <1>       mov    cx, 3
0000D1AE 6689C3              <1>       mov    bx, ax
0000D1B1 6689C8              <1>       mov    ax, cx ; 3
0000D1B4 66F7E2              <1>       mul    dx     ; Multiply by 3
0000D1B7 66D1E8              <1>       shr    ax, 1  ; Divide by 2
0000D1BA 6693                <1>       xchg   bx, ax
                             <1>       ; EAX = Count of 3 FAT sectors
                             <1>       ; EBX = Byte Offset in FAT buffer
0000D1BC 66F7E1              <1>       mul    cx  ; 3 * AX
                             <1>       ; EAX = FAT Beginning Sector
                             <1>       ; EDX = 0
0000D1BF 8A0D[F2910100]      <1>       mov    cl, [FAT_BuffValidData]
                             <1>       ; TRDOS v1 has a FATal bug here !
                             <1>       ; (it does not have 'cmp cl, 2' instruction here !
                             <1>       ;  while 'jne' is existing !)
0000D1C5 80F902              <1>       cmp    cl, 2 ; 2 = dirty buffer (must be written to disk)
0000D1C8 750A                <1>       jne    short loc_uc_check_fat12_buff_sector_load
                             <1>
                             <1> loc_uc_check_fat12_buff_sector_save:
0000D1CA 3B05[F6910100]      <1>       cmp    eax, [FAT_BuffSector]
0000D1D0 75B2                <1>         jne    short loc_uc_save_fat_buffer
0000D1D2 EB15                <1>       jmp    short loc_update_fat12_cell
                             <1>
                             <1> loc_uc_check_fat12_buff_sector_load:
0000D1D4 80F901              <1>       cmp    cl, 1 ; byte ptr [FAT_BuffValidData]
0000D1D7 0F8550010000        <1>         jne    loc_uc_load_fat_sectors
0000D1DD 3B05[F6910100]      <1>       cmp    eax, [FAT_BuffSector]
0000D1E3 0F8544010000        <1>         jne    loc_uc_load_fat_sectors
                             <1>
                             <1> loc_update_fat12_cell:
0000D1E9 81C3001C0900        <1>       add    ebx, FAT_Buffer ; 26/02/2016
0000D1EF 668B0D[EE910100]    <1>       mov    cx, [FAT_CurrentCluster]
0000D1F6 66D1E9              <1>       shr    cx, 1
0000D1F9 668B03              <1>       mov    ax, [ebx]
0000D1FC 6689C2              <1>       mov    dx, ax
0000D1FF 7344                <1>       jnc    short uc_fat12_nc_even
                             <1>
0000D201 6683E00F            <1>       and    ax, 0Fh
0000D205 8B0D[90940100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
0000D20B 66C1E104            <1>       shl    cx, 4
0000D20F 6609C1              <1>       or     cx, ax
0000D212 6689D0              <1>       mov    ax, dx
0000D215 66890B              <1>       mov    [ebx], cx  ; 16 bits !
0000D218 66C1E804            <1>       shr    ax, 4 ; al(bit4..7)+ah(bit0..7)
                             <1>
                             <1> update_fat12_buffer:
0000D21C A3[EE910100]        <1>       mov    [FAT_CurrentCluster], eax
0000D221 89C2                <1>       mov    edx, eax ; 01/03/2016
0000D223 C605[F2910100]02    <1>       mov    byte [FAT_BuffValidData], 2
0000D22A 6683F802            <1>       cmp    ax, 2
0000D22E 0F826AFFFFFF        <1>         jb     return_uc_fat_stc
0000D234 3B05[FE910100]      <1>       cmp    eax, [LastCluster]
0000D23A 0F875EFFFFFF        <1>         ja     return_uc_fat_stc
0000D240 E927FFFFFF          <1>         jmp    loc_fat_buffer_updated
                             <1>
                             <1> uc_fat12_nc_even:
0000D245 662500F0            <1>       and    ax, 0F000h
0000D249 8B0D[90940100]      <1>       mov    ecx, [ClusterValue] ; 32 bits
0000D24F 80E50F              <1>       and    ch, 0Fh
0000D252 6609C1              <1>       or     cx, ax
0000D255 6689D0              <1>       mov    ax, dx
0000D258 66890B              <1>       mov    [ebx], cx ; 16 bits !
0000D25B 80E40F              <1>       and    ah, 0Fh ; al(bit0..7)+ah(bit0..3)
0000D25E EBBC                <1>       jmp    short update_fat12_buffer
                             <1>
                             <1> update_fat32_cluster:
0000D260 8B4E78              <1>       mov    ecx, [esi+LD_Clusters]
0000D263 41                  <1>       inc    ecx
0000D264 890D[FE910100]      <1>       mov    [LastCluster], ecx
                             <1>
```

```
0000D26A 39C8              <1>       cmp    eax, ecx
0000D26C 0F872CFFFFFF      <1>         ja      return_uc_fat_stc
                           <1>
                           <1> pass_uc_fat32_errc:
                           <1>       ;sub   edx, edx
0000D272 BB80010000        <1>       mov    ebx, 180h ;384
0000D277 F7F3              <1>       div    ebx
                           <1>       ; EAX = Count of 3 FAT sectors
                           <1>       ; DX = Cluster offset in FAT buffer
0000D279 89D3              <1>       mov    ebx, edx
0000D27B C1E302            <1>       shl    ebx, 2 ; Multiply by 4
0000D27E BA03000000        <1>       mov    edx, 3
0000D283 F7E2              <1>       mul    edx
                           <1>       ; EBX = Cluster Offset in FAT buffer
                           <1>       ; EAX = FAT Sector
                           <1>       ; EDX = 0
0000D285 8A0D[F2910100]    <1>       mov    cl, [FAT_BuffValidData]
0000D28B 80F902            <1>       cmp    cl, 2
0000D28E 750E              <1>       jne    short loc_uc_check_fat32_buff_sector_load
                           <1>
                           <1> loc_uc_check_fat32_buff_sector_save:
0000D290 3B05[F6910100]    <1>       cmp    eax, [FAT_BuffSector]
0000D296 0F85E8FEFFFF      <1>         jne     loc_uc_save_fat_buffer
0000D29C EB11              <1>       jmp    short loc_update_fat32_cell
                           <1>
                           <1> loc_uc_check_fat32_buff_sector_load:
0000D29E 80F901            <1>       cmp    cl, 1 ; byte [FAT_BuffValidData]
0000D2A1 0F8586000000      <1>         jne     loc_uc_load_fat_sectors
0000D2A7 3B05[F6910100]    <1>       cmp    eax, [FAT_BuffSector]
0000D2AD 757E              <1>         jne     loc_uc_load_fat_sectors
                           <1>
                           <1> loc_update_fat32_cell:
                           <1> loc_update_fat32_buffer:
0000D2AF 81C3001C0900      <1>       add    ebx, FAT_Buffer ; 26/02/2016
0000D2B5 8B03              <1>       mov    eax, [ebx]
0000D2B7 25FFFFFF0F        <1>       and    eax, 0FFFFFFFh ; 28 bit cluster value
                           <1>
0000D2BC 8B15[EE910100]    <1>       mov    edx, [FAT_CurrentCluster] ; 01/03/2016
                           <1>
0000D2C2 A3[EE910100]      <1>       mov    [FAT_CurrentCluster], eax
0000D2C7 8B0D[90940100]    <1>       mov    ecx, [ClusterValue]
0000D2CD 890B              <1>       mov    [ebx], ecx ; 29/02/2016
                           <1>
0000D2CF C605[F2910100]02  <1>       mov    byte [FAT_BuffValidData], 2
                           <1>
                           <1>       ; 01/03/2016
0000D2D6 21C0              <1>       and    eax, eax ; was it free cluster ?
0000D2D8 7514              <1>       jnz    short loc_upd_fat32_c0
                           <1>
                           <1>       ;or    ecx, ecx ; it will be left free ?!
                           <1>       ;jz    short loc_upd_fat32_c3
                           <1>
0000D2DA 3B563E            <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
0000D2DD 7520              <1>       jne    short loc_upd_fat32_c3
                           <1>
0000D2DF 3B15[FE910100]    <1>       cmp    edx, [LastCluster]
0000D2E5 7207              <1>       jb     short loc_upd_fat32_c0
                           <1>
0000D2E7 BA02000000        <1>       mov    edx, 2 ; rewind !
0000D2EC EB0E              <1>       jmp    short loc_upd_fat32_c2
                           <1>
                           <1> loc_upd_fat32_c0:
0000D2EE FF463E            <1>       inc    dword [esi+LD_BPB+BPB_Reserved+4] ; set it to next cluster
0000D2F1 EB0C              <1>       jmp    short loc_upd_fat32_c3
                           <1>
                           <1> loc_upd_fat32_c1:
0000D2F3 09C9              <1>       or     ecx, ecx ; will it be free cluster ?
0000D2F5 7508              <1>       jnz    short loc_upd_fat32_c3
                           <1>
0000D2F7 3B563E            <1>       cmp    edx, [esi+LD_BPB+BPB_Reserved+4] ; First free cluster
0000D2FA 7303              <1>       jnb    short loc_upd_fat32_c3
                           <1>
                           <1> loc_upd_fat32_c2:
0000D2FC 89563E            <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
                           <1>
                           <1> loc_upd_fat32_c3:
0000D2FF 89C2              <1>       mov    edx, eax
                           <1>
                           <1> loc_upd_fat32_c4:
0000D301 83F802            <1>       cmp    eax, 2
0000D304 0F8294FEFFFF      <1>         jb      return_uc_fat_stc
                           <1>
                           <1> pass_uc_fat32_c_zero_check_2:
0000D30A 3B05[FE910100]    <1>       cmp    eax, [LastCluster]
0000D310 0F8788FEFFFF      <1>         ja      return_uc_fat_stc
                           <1>
0000D316 E951FEFFFF        <1>       jmp     loc_fat_buffer_updated
                           <1>
                           <1> loc_fat_sectors_rw_error1:
                           <1>       ;mov   byte [FAT_BuffValidData], 0
                           <1>       ; 23/10/2016 (15h -> 17)
                           <1>       ; 23/03/2016
0000D31B B811000000        <1>       mov    eax, 17 ; Drive not ready or read error
0000D320 8825[F2910100]    <1>       mov    [FAT_BuffValidData], ah ; 0
                           <1>
                           <1> loc_fat_sectors_rw_error2:
                           <1>       ;mov   eax, error code
                           <1>       ;mov   edx, 0
0000D326 8B0D[90940100]    <1>       mov    ecx, [ClusterValue]
0000D32C C3                <1>       retn
                           <1>
                           <1> loc_uc_load_fat_sectors:
0000D32D A3[F6910100]      <1>       mov    [FAT_BuffSector], eax
                           <1>
                           <1> load_uc_fat_sectors_zero:
```

```
 976 0000D332 034660              <1>       add    eax, [esi+LD_FATBegin]
 977 0000D335 BB001C0900          <1>       mov    ebx, FAT_Buffer
 978 0000D33A B903000000          <1>       mov    ecx, 3
 979 0000D33F E85C580000          <1>       call   disk_read
 980 0000D344 72D5                <1>       jc     short loc_fat_sectors_rw_error1
 981                              <1>
 982 0000D346 C605[F2910100]01    <1>         mov     byte [FAT_BuffValidData], 1
 983 0000D34D A1[EE910100]        <1>       mov    eax, [FAT_CurrentCluster]
 984 0000D352 8B0D[90940100]      <1>       mov    ecx, [ClusterValue]
 985 0000D358 E972FDFFFF          <1>         jmp     loc_update_cluster_check_fat_type
 986                              <1>
 987                              <1> save_fat_buffer:
 988                              <1>       ; 15/10/2016
 989                              <1>       ; 01/03/2016
 990                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
 991                              <1>       ; 11/08/2011
 992                              <1>       ; 09/02/2005
 993                              <1>       ; INPUT ->
 994                              <1>       ;      None
 995                              <1>       ; OUTPUT ->
 996                              <1>       ;      cf = 0 -> OK.
 997                              <1>       ;      cf = 1 -> error code in AL (EAX)
 998                              <1>       ;
 999                              <1>       ;      EBX = FAT_Buffer address
1000                              <1>       ;
1001                              <1>       ; (EAX, EDX, ECX will be modified)
1002                              <1>
1003                              <1>       ;cmp   byte [FAT_BuffValidData], 2
1004                              <1>       ;je    short loc_save_fat_buff
1005                              <1>
1006                              <1> ;loc_save_fat_buffer_retn:
1007                              <1> ;     xor    eax, eax
1008                              <1> ;     retn
1009                              <1>
1010                              <1> loc_save_fat_buff:
1011 0000D35D 31D2                <1>       xor    edx, edx
1012 0000D35F 8A35[F3910100]      <1>       mov    dh, [FAT_BuffDrvName]
1013 0000D365 80FE41              <1>       cmp    dh, 'A'
1014 0000D368 722E                <1>       jb     short loc_save_fat_buffer_inv_data_retn
1015 0000D36A 80EE41              <1>       sub    dh, 'A'
1016 0000D36D 56                  <1>       push   esi ; *
1017 0000D36E BE00010900          <1>         mov     esi, Logical_DOSDisks
1018 0000D373 01D6                <1>       add    esi, edx
1019                              <1>
1020 0000D375 8A5603              <1>       mov    dl, [esi+LD_FATType]
1021 0000D378 20D2                <1>       and    dl, dl
1022 0000D37A 741B                <1>       jz     short loc_save_fat_buffer_inv_data_pop_retn
1023                              <1>
1024 0000D37C A1[F6910100]        <1>       mov    eax, [FAT_BuffSector]
1025 0000D381 80FA02              <1>       cmp    dl, 2
1026 0000D384 770A                <1>       ja     short loc_save_fat32_buff
1027                              <1>
1028                              <1> loc_save_fat_12_16_buff:
1029                              <1>       ; 01/03/2016
1030                              <1>       ; TRDOS v1 has a FATal bug here!
1031                              <1>       ; Correct code: mov dx, word ptr [FAT_BuffSector]+2
1032                              <1>       ; (DX:AX in TRDOS v1 -> EAX in TRDOS v2)
1033                              <1>       ;
1034 0000D386 0FB74E1C            <1>       movzx  ecx, word [esi+LD_BPB+FATSecs]
1035 0000D38A 29C1                <1>       sub    ecx, eax
1036                              <1>       ; TRDOS v1 has a bug here... ('pop esi' was forgotten!)
1037                              <1>       ;jna   short loc_save_fat_buffer_inv_data_retn ; wrong addr!
1038 0000D38C 7609                <1>       jna    short loc_save_fat_buffer_inv_data_pop_retn ; correct addr.
1039 0000D38E EB15                <1>       jmp    short loc_save_fat_buffer_check_rs3
1040                              <1>
1041                              <1> loc_save_fat32_buff:
1042 0000D390 8B4E2A              <1>       mov    ecx, [esi+LD_BPB+FAT32_FAT_Size]
1043 0000D393 29C1                <1>       sub    ecx, eax
1044 0000D395 770E                <1>       ja     short loc_save_fat_buffer_check_rs3
1045                              <1>
1046                              <1> loc_save_fat_buffer_inv_data_pop_retn:
1047 0000D397 5E                  <1>       pop    esi ; *
1048                              <1> loc_save_fat_buffer_inv_data_retn:
1049 0000D398 B80D000000          <1>       mov    eax, 0Dh ; Invalid DATA
1050 0000D39D C3                  <1>       retn
1051                              <1>
1052                              <1> loc_save_fat_buff_remain_sectors_3:
1053 0000D39E B903000000          <1>       mov    ecx, 3
1054 0000D3A3 EB05                <1>       jmp    short loc_save_fat_buff_continue
1055                              <1>
1056                              <1> loc_save_fat_buffer_check_rs3:
1057 0000D3A5 83F903              <1>       cmp    ecx, 3
1058 0000D3A8 77F4                <1>       ja     short loc_save_fat_buff_remain_sectors_3
1059                              <1>
1060                              <1> loc_save_fat_buff_continue:
1061 0000D3AA BB001C0900          <1>       mov    ebx, FAT_Buffer
1062 0000D3AF 034660              <1>       add    eax, [esi+LD_FATBegin]
1063 0000D3B2 51                  <1>       push   ecx
1064 0000D3B3 E8D9570000          <1>       call   disk_write
1065 0000D3B8 59                  <1>       pop    ecx
1066 0000D3B9 722B                <1>       jc     short loc_save_FAT_buff_write_err
1067                              <1>
1068 0000D3BB 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
1069 0000D3BF 7605                <1>       jna    short loc_calc_2nd_fat12_16_addr
1070                              <1>
1071                              <1> loc_calc_2nd_fat32_addr:
1072 0000D3C1 8B462A              <1>       mov    eax, [esi+LD_BPB+FAT32_FAT_Size]
1073 0000D3C4 EB04                <1>       jmp    short loc_calc_2nd_fat_addr
1074                              <1>
1075                              <1> loc_calc_2nd_fat12_16_addr:
1076 0000D3C6 0FB7461C            <1>       movzx  eax, word [esi+LD_BPB+FATSecs]
1077                              <1>
1078                              <1> loc_calc_2nd_fat_addr:
1079 0000D3CA 034660              <1>       add    eax, [esi+LD_FATBegin]
1080 0000D3CD 0305[F6910100]      <1>       add    eax, [FAT_BuffSector]
```

```
1081 0000D3D3 BB001C0900          <1>       mov    ebx, FAT_Buffer
1082                              <1>       ; ecx = 1 to 3
1083 0000D3D8 E8B4570000          <1>       call   disk_write
1084 0000D3DD 7207                <1>       jc     short loc_save_FAT_buff_write_err
1085                              <1>       ; Valid  buffer (1 = valid but do not save)
1086 0000D3DF C605[F2910100]01    <1>       mov    byte [FAT_BuffValidData], 1
1087                              <1>
1088                              <1> loc_save_FAT_buff_write_err:
1089 0000D3E6 5E                  <1>       pop    esi ; *
1090 0000D3E7 BB001C0900          <1>       mov    ebx, FAT_Buffer
1091                              <1>       ; 15/10/2016 (1Dh -> 18)
1092                              <1>       ; 23/03/2016 (1Dh)
1093 0000D3EC B812000000          <1>       mov    eax, 18 ; Drive not ready or write error
1094 0000D3F1 C3                  <1>       retn
1095                              <1>
1096                              <1> calculate_fat_freespace:
1097                              <1>       ; 23/03/2016
1098                              <1>       ; 02/03/2016
1099                              <1>       ; 01/03/2016
1100                              <1>       ; 29/02/2016
1101                              <1>       ; 22/02/2016 (TRDOS 386 =  TRDOS v2.0)
1102                              <1>       ; 30/04/2011
1103                              <1>       ; 03/04/2010
1104                              <1>       ; 2005
1105                              <1>       ; INPUT ->
1106                              <1>       ;      EAX = Cluster count to be added or subtracted
1107                              <1>       ;      If BH = FFh, ESI = TR-DOS Logical Drive Description Table
1108                              <1>       ;      If BH < FFh, BH = TR-DOS Logical Drive Number
1109                              <1>       ;      BL:
1110                              <1>       ;      0 = Calculate, 1 = Add, 2 = Subtract, 3 = Get (Not Set/Calc)
1111                              <1>       ; OUTPUT ->
1112                              <1>       ;      EAX = Free Space in sectors
1113                              <1>       ;      ESI = Logical Dos Drive Description Table address
1114                              <1>       ;      BH = Logical Dos Drive Number (same with input value of BH)
1115                              <1>       ;      BL = Type of operation (same with input value of BL)
1116                              <1>       ;      ECX = 0 -> valid
1117                              <1>       ;      ECX > 0 -> error or invalid
1118                              <1>       ;      If EAX = FFFFFFFFh, it is 're-calculation needed'
1119                              <1>       ;                              sign due to r/w error
1120                              <1>
1121 0000D3F2 66891D[96940100]    <1>       mov    [CFS_OPType], bx
1122 0000D3F9 A3[98940100]        <1>       mov    [CFS_CC], eax
1123                              <1>
1124 0000D3FE 80FFFF              <1>       cmp    bh, 0FFh
1125 0000D401 740B                <1>       je     short pass_calculate_freespace_get_drive_dt_offset
1126                              <1>
1127                              <1> loc_calculate_freespace_get_drive_dt_offset:
1128 0000D403 31C0                <1>       xor    eax, eax
1129 0000D405 88FC                <1>         mov     ah, bh
1130 0000D407 BE00010900          <1>       mov    esi, Logical_DOSDisks
1131 0000D40C 01C6                <1>         add     esi, eax
1132                              <1>
1133                              <1> pass_calculate_freespace_get_drive_dt_offset:
1134 0000D40E 08DB                <1>       or     bl, bl
1135 0000D410 7435                <1>       jz     short loc_reset_fcc
1136                              <1>
1137                              <1> loc_get_free_sectors:
1138 0000D412 8B4674              <1>       mov    eax, [esi+LD_FreeSectors]
1139                              <1>
1140                              <1>       ;xor   ecx, ecx
1141                              <1>       ;dec   ecx ; 0FFFFFFFFh
1142                              <1>       ;cmp   eax, ecx ; 29/02/2016
1143                              <1>       ;je    short loc_get_free_sectors_retn ; recalculation is needed!
1144                              <1>
1145                              <1>       ; 23/03/2016
1146 0000D415 8B4E70              <1>       mov    ecx, [esi+LD_TotalSectors]
1147 0000D418 39C1                <1>       cmp    ecx, eax ; Total sectors must be greater than Free sectors !
1148 0000D41A 7707                <1>       ja     short loc_get_free_sectors_check_optype
1149                              <1>
1150 0000D41C 31C0                <1>       xor    eax, eax
1151 0000D41E 48                  <1>       dec    eax ; 0FFFFFFFFh  ; recalculation is needed!
1152 0000D41F 894674              <1>       mov    [esi+LD_FreeSectors], eax ; reset (for recalculation)
1153                              <1>
1154                              <1> loc_get_free_sectors_retn:
1155 0000D422 C3                  <1>       retn
1156                              <1>
1157                              <1> loc_get_free_sectors_check_optype:
1158 0000D423 80FB03              <1>       cmp    bl, 3
1159 0000D426 7203                <1>       jb     short loc_set_fcc
1160                              <1>
1161 0000D428 29C9                <1>       sub    ecx, ecx ; 0
1162                              <1>
1163 0000D42A C3                  <1>       retn
1164                              <1>
1165                              <1> loc_set_fcc:
1166 0000D42B 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
1167 0000D42F 0F87DF000000        <1>         ja      loc_update_FAT32_fs_info_fcc
1168                              <1>
1169                              <1>       ;mov   eax, [esi+LD_FreeSectors]
1170 0000D435 0FB64E13            <1>       movzx  ecx, byte [esi+LD_BPB+SecPerClust]
1171 0000D439 29D2                <1>       sub    edx, edx
1172 0000D43B F7F1                <1>       div    ecx
1173                              <1>       ;or    dx, dx
1174                              <1>       ;      ; DX -> Remain sectors < SecPerClust
1175                              <1>       ;      ; DX > 0 -> invalid free sector count
1176                              <1>       ;jnz   short loc_reset_fcc
1177                              <1>
1178                              <1> ;pass_set_fcc_div32:
1179 0000D43D A3[0F920100]        <1>       mov    [FreeClusterCount], eax
1180 0000D442 E988000000          <1>         jmp     loc_set_free_sectors_FAT12_FAT16
1181                              <1>
1182                              <1> loc_reset_fcc:
1183 0000D447 31C0                <1>       xor    eax, eax
1184 0000D449 A3[0F920100]        <1>       mov    [FreeClusterCount], eax ; 0
1185 0000D44E 8B5678              <1>       mov    edx, [esi+LD_Clusters]
```

```
1186 0000D451 42                  <1>       inc    edx
1187 0000D452 8915[FE910100]      <1>       mov    [LastCluster], edx
1188                              <1>
1189 0000D458 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
1190 0000D45C 7647                <1>       jna    short loc_count_free_fat_clusters_0
1191                              <1>
1192 0000D45E 48                  <1>       dec    eax ; FFFFFFFFh
1193 0000D45F A3[A0940100]        <1>       mov    [CFS_FAT32FC], eax
1194                              <1>
1195                              <1>       ; 29/02/2016
1196 0000D464 89463A              <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; reset
1197 0000D467 89463E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], eax ; reset
1198                              <1>
1199 0000D46A B802000000          <1>       mov    eax, 2
1200                              <1>
1201                              <1> loc_count_fc_next_cluster_0:
1202 0000D46F 50                  <1>       push   eax
1203 0000D470 E801F9FFFF          <1>       call   get_next_cluster
1204 0000D475 7310                <1>       jnc    short loc_check_fat32_ff_cluster
1205 0000D477 09C0                <1>       or     eax, eax
1206 0000D479 741E                <1>       jz     short pass_inc_cfs_fcc_0
1207                              <1>
1208                              <1> loc_put_fcc_unknown_sign:
1209 0000D47B 58                  <1>       pop    eax
1210                              <1>       ; "Free count is Unknown" sign
1211                              <1>       ;mov   dword [FreeClusterCount], 0FFFFFFFFh
1212                              <1>
1213                              <1>       ; 29/02/2016
1214                              <1>       ; Save Free Cluster Count value in FAT32 'BPB_Reserved' area
1215                              <1>       ;mov   [esi+LD_BPB+BPB_Reserved], 0FFFFFFFFh ; unknown!
1216 0000D47C 8B15[A0940100]      <1>       mov    edx, [CFS_FAT32FC] ; First Free Cluster
1217                              <1>       ; Save First Free Cluster value in FAT32 'BPB_Reserved+4' area
1218 0000D482 89563E              <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx
1219                              <1>
1220 0000D485 EB7D                <1>         jmp     loc_put_fcc_invalid_sign
1221                              <1>
1222                              <1> loc_check_fat32_ff_cluster:
1223 0000D487 09C0                <1>       or     eax, eax
1224 0000D489 750E                <1>       jnz    short pass_inc_cfs_fcc_0
1225 0000D48B 58                  <1>       pop    eax
1226 0000D48C A3[A0940100]        <1>       mov    [CFS_FAT32FC], eax
1227                              <1>       ;mov   dword [FreeClusterCount], 1
1228 0000D491 FF05[0F920100]      <1>       inc    dword [FreeClusterCount]
1229 0000D497 EB27                <1>       jmp    short pass_inc_cfs_fcc_1
1230                              <1>
1231                              <1> pass_inc_cfs_fcc_0:
1232 0000D499 58                  <1>       pop    eax
1233                              <1>
1234                              <1> pass_inc_cfs_fcc_0c:
1235 0000D49A 40                  <1>       inc    eax ; add eax, 1
1236 0000D49B 3B05[FE910100]      <1>       cmp    eax, [LastCluster]
1237 0000D4A1 76CC                <1>       jna    short loc_count_fc_next_cluster_0
1238 0000D4A3 EB6F                <1>       jmp    short loc_update_FAT32_fs_info_fcc
1239                              <1>
1240                              <1> loc_count_free_fat_clusters_0:
1241                              <1>       ;mov   eax, 2
1242 0000D4A5 B002                <1>       mov    al, 2
1243                              <1>
1244                              <1> loc_count_fc_next_cluster:
1245 0000D4A7 50                  <1>       push   eax
1246 0000D4A8 E8C9F8FFFF          <1>       call   get_next_cluster
1247 0000D4AD 720C                <1>       jc     short loc_count_fcc_stc
1248                              <1>
1249                              <1> loc_count_free_clusters_1:
1250 0000D4AF 21C0                <1>       and    eax, eax
1251 0000D4B1 750C                <1>       jnz    short pass_inc_cfs_fcc
1252                              <1>
1253 0000D4B3 FF05[0F920100]      <1>       inc    dword [FreeClusterCount]
1254 0000D4B9 EB04                <1>       jmp    short pass_inc_cfs_fcc
1255                              <1>
1256                              <1> loc_count_fcc_stc:
1257 0000D4BB 09C0                <1>       or     eax, eax
1258 0000D4BD 75BC                <1>       jnz    short loc_put_fcc_unknown_sign ; 29/02/2016
1259                              <1>
1260                              <1> pass_inc_cfs_fcc:
1261 0000D4BF 58                  <1>       pop    eax
1262                              <1>
1263                              <1> pass_inc_cfs_fcc_1:
1264 0000D4C0 40                  <1>       inc    eax ; add eax, 1
1265 0000D4C1 3B05[FE910100]      <1>       cmp    eax, [LastCluster]
1266 0000D4C7 76DE                <1>       jna    short loc_count_fc_next_cluster
1267                              <1>
1268                              <1> loc_set_free_sectors:
1269 0000D4C9 807E0302            <1>       cmp    byte [esi+LD_FATType], 2
1270 0000D4CD 7745                <1>       ja     short loc_update_FAT32_fs_info_fcc
1271                              <1>
1272                              <1> loc_set_free_sectors_FAT12_FAT16:
1273 0000D4CF 803D[96940100]00    <1>       cmp    byte [CFS_OPType], 0
1274 0000D4D6 761C                <1>       jna    short pass_FAT_add_sub_fcc
1275 0000D4D8 A1[98940100]        <1>       mov    eax, [CFS_CC]
1276 0000D4DD 803D[96940100]01    <1>       cmp    byte [CFS_OPType], 1
1277 0000D4E4 7708                <1>       ja     short pass_FAT_add_fcc
1278 0000D4E6 0105[0F920100]      <1>       add    [FreeClusterCount], eax
1279 0000D4EC EB06                <1>       jmp    short pass_FAT_add_sub_fcc
1280                              <1>
1281                              <1> pass_FAT_add_fcc:
1282 0000D4EE 2905[0F920100]      <1>       sub    [FreeClusterCount], eax
1283                              <1>
1284                              <1> pass_FAT_add_sub_fcc:
1285 0000D4F4 0FB64613            <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
1286 0000D4F8 8B15[0F920100]      <1>       mov    edx, [FreeClusterCount]
1287 0000D4FE F7E2                <1>       mul    edx
1288                              <1>
1289 0000D500 31C9                <1>       xor    ecx, ecx
1290 0000D502 EB05                <1>       jmp    short loc_cfs_retn_params
```

```
1291                                    <1>
1292                                    <1> loc_put_fcc_invalid_sign:
1293 0000D504 29C0                     <1>             sub    eax, eax ; 0
1294 0000D506 48                       <1>       dec    eax ; FFFFFFFFh
1295                                    <1> loc_fat32_ffc_recalc_needed:
1296 0000D507 89C1                     <1>       mov    ecx, eax
1297                                    <1>
1298                                    <1> loc_cfs_retn_params:
1299 0000D509 894674                   <1>       mov    [esi+LD_FreeSectors], eax
1300 0000D50C 0FB71D[96940100]         <1>       movzx  ebx, word [CFS_OPType]
1301 0000D513 C3                       <1>       retn
1302                                    <1>
1303                                    <1> loc_update_FAT32_fs_info_fcc:
1304                                    <1> loc_check_fcc_FSINFO_op:
1305                                    <1>       ; 29/02/2016
1306                                    <1>       ; EAX = Free cluster count (before this update) ; value from disk
1307                                    <1>       ; EDX = First Free Cluster (before this update) ; value from disk
1308 0000D514 803D[96940100]01         <1>       cmp    byte [CFS_OPType], 1
1309 0000D51B 7221                     <1>       jb     short loc_cfs_FAT32_get_rcalc_parms ; 0 = recalculated
1310 0000D51D 7406                     <1>       je     short loc_check_fcc_FSINFO_op1 ; 1 = add
1311                                    <1> loc_check_fcc_FSINFO_op2: ; subtract
1312 0000D51F F71D[98940100]           <1>       neg    dword [CFS_CC] ; prepare to subtract ; 2 = sub (add negative)
1313                                    <1> loc_check_fcc_FSINFO_op1:
1314                                    <1>       ; 01/03/2016
1315 0000D525 31D2                     <1>       xor    edx, edx ; 0
1316 0000D527 4A                       <1>       dec    edx ; 0FFFFFFFFh
1317 0000D528 8B463A                   <1>       mov    eax, [esi+LD_BPB+BPB_Reserved]
1318 0000D52B 39D0                     <1>       cmp    eax, edx
1319 0000D52D 73D5                     <1>       jnb    short loc_put_fcc_invalid_sign
1320 0000D52F 0305[98940100]           <1>         add     eax, [CFS_CC] ; free cluster count on disk + current count
1321 0000D535 72CD                     <1>       jc     short loc_put_fcc_invalid_sign
1322                                    <1>
1323 0000D537 A3[0F920100]             <1>       mov    [FreeClusterCount], eax
1324 0000D53C EB0E                     <1>       jmp    short loc_cfs_write_FSINFO_sector
1325                                    <1>
1326                                    <1> loc_cfs_FAT32_get_rcalc_parms:
1327 0000D53E 8B15[A0940100]           <1>       mov    edx, [CFS_FAT32FC]
1328 0000D544 A1[0F920100]             <1>       mov    eax, [FreeClusterCount]
1329 0000D549 89563E                   <1>       mov    [esi+LD_BPB+BPB_Reserved+4], edx ; First Free Cluster
1330                                    <1> loc_cfs_write_FSINFO_sector:
1331 0000D54C 89463A                   <1>       mov    [esi+LD_BPB+BPB_Reserved], eax ; Free cluster count
1332                                    <1>       ; 01/03/2016
1333 0000D54F E8AA000000               <1>       call   set_fat32_fsinfo_sector_parms
1334 0000D554 72AE                     <1>         jc      short loc_put_fcc_invalid_sign
1335                                    <1>
1336                                    <1> loc_set_FAT32_free_sectors:
1337                                    <1>       ; 29/02/2016
1338                                    <1>       ;mov   eax, [FreeClusterCount]
1339                                    <1>       ;mov   ecx, eax
1340                                    <1>       ;cmp   eax, 0FFFFFFFFh ; Invalid !
1341                                    <1>       ;je    short loc_cfs_retn_params
1342                                    <1>       ;
1343 0000D556 8B0D[0F920100]             <1>       mov    ecx, [FreeClusterCount]
1344 0000D55C 0FB64613                 <1>       movzx  eax, byte [esi+LD_BPB+SecPerClust]
1345 0000D560 F7E1                     <1>       mul    ecx
1346                                    <1>       ; 29/02/2016
1347 0000D562 31C9                     <1>       xor    ecx, ecx ; 0
1348 0000D564 09D2                     <1>       or     edx, edx ; 0 ?
1349 0000D566 759C                     <1>         jnz     loc_put_fcc_invalid_sign
1350 0000D568 394670                   <1>       cmp    [esi+LD_TotalSectors], eax ; Volume size in sectors
1351 0000D56B 7697                     <1>         jna     short loc_put_fcc_invalid_sign
1352                                    <1>       ;
1353                                    <1> loc_set_FAT32_free_sectors_ok:
1354 0000D56D 31D2                     <1>       xor    edx, edx ; 0
1355 0000D56F EB98                     <1>         jmp     short loc_cfs_retn_params
1356                                    <1>       ;
1357                                    <1>
1358                                    <1> get_last_cluster:
1359                                    <1>       ; 22/10/2016
1360                                    <1>       ; 27/02/2016 (TRDOS 386 =  TRDOS v2.0)
1361                                    <1>       ; 12/06/2010 (DRV_FAT.ASM, 'proc_get_last_custer')
1362                                    <1>       ; 06/06/2010
1363                                    <1>       ; INPUT ->
1364                                    <1>       ;      EAX = First Cluster Number
1365                                    <1>       ;      ESI = Logical Dos Drive Parameters Table
1366                                    <1>       ; OUTPUT ->
1367                                    <1>       ;      cf = 0 -> No Error, EAX is valid
1368                                    <1>       ;      cf = 1 -> EAX > 0 -> Error
1369                                    <1>       ;      EAX = Last Cluster Number
1370                                    <1>       ;       ECX = Previous Cluster -just before the last cluster-
1371                                    <1>       ;       ; 22/10/2016
1372                                    <1>       ;      [glc_index] = cluster index number of the last cluster
1373                                    <1>       ;
1374                                    <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
1375                                    <1>
1376 0000D571 89C1                     <1>       mov    ecx, eax
1377                                    <1>
1378 0000D573 C705[A8940100]FFFF-       <1>       mov    dword [glc_index], 0FFFFFFFFh ; 22/10/2016
1378 0000D57B FFFF                     <1>
1379                                    <1>
1380                                    <1> loc_glc_get_next_cluster_1:
1381 0000D57D 890D[A4940100]           <1>       mov    [glc_prevcluster], ecx
1382                                    <1>       ; 22/10/2016
1383 0000D583 FF05[A8940100]           <1>       inc    dword [glc_index]
1384                                    <1>
1385                                    <1> loc_glc_get_next_cluster_2:
1386 0000D589 E8E8F7FFFF               <1>       call   get_next_cluster
1387                                    <1>       ; ecx = current/previous cluster
1388                                    <1>       ; eax = next/last cluster
1389 0000D58E 73ED                     <1>       jnc    short loc_glc_get_next_cluster_1
1390                                    <1>
1391 0000D590 09C0                     <1>       or     eax, eax
1392 0000D592 7509                     <1>       jnz    short loc_glc_stc_retn
1393                                    <1>
1394                                    <1>       ; ecx = previous cluster
```

```
1395 0000D594 89C8                <1>         mov  eax, ecx
1396                              <1>
1397                              <1>       ; previous cluster becomes last cluster (ecx -> eax)
1398                              <1>       ; previous of previous cluster becomes previous cluster (ecx)
1399                              <1>
1400                              <1> loc_glc_prev_cluster_retn:
1401 0000D596 8B0D[A4940100]      <1>       mov   ecx, [glc_prevcluster]
1402 0000D59C C3                  <1>       retn
1403                              <1>
1404                              <1> loc_glc_stc_retn:
1405 0000D59D F5                  <1>       cmc   ;stc
1406 0000D59E EBF6                <1>         jmp  short loc_glc_prev_cluster_retn
1407                              <1>
1408                              <1> truncate_cluster_chain:
1409                              <1>       ; 01/03/2016
1410                              <1>       ; 28/02/2016 (TRDOS 386 =  TRDOS v2.0)
1411                              <1>       ; 22/01/2011 (DRV_FAT.ASM, 'proc_truncate_cluster_chain')
1412                              <1>       ; 11/09/2010
1413                              <1>       ; INPUT ->
1414                              <1>       ;     ESI = Logical dos drive description table address
1415                              <1>       ;     EAX = First cluster to be truncated/unlinked
1416                              <1>       ; OUTPUT ->
1417                              <1>       ;     ESI = Logical dos drive description table address
1418                              <1>       ;     ECX = Count of truncated/removed clusters
1419                              <1>       ;     CF = 0 -> EAX = Free sectors
1420                              <1>       ;     CF = 1 -> Error code in EAX (AL)
1421                              <1>
1422                              <1>       ; NOTE: This procedure does not update lm date&time !
1423                              <1>
1424                              <1> loc_truncate_cc:
1425 0000D5A0 31C9                <1>       xor   ecx, ecx ; mov ecx, 0
1426                              <1>       ;mov  byte [FAT_BuffValidData], 0
1427 0000D5A2 890D[FA910100]      <1>       mov   [FAT_ClusterCounter], ecx ; 0 ; reset
1428                              <1>
1429                              <1> loc_tcc_unlink_clusters:
1430 0000D5A8 E8F3FAFFFF          <1>       call  update_cluster
1431                              <1>       ; EAX = Next Cluster
1432                              <1>       ; ECX = Cluster Value
1433                              <1>       ; Note:
1434                              <1>       ; Returns count of unlinked clusters in
1435                              <1>       ; dword ptr FAT_ClusterCounter
1436 0000D5AD 73F9                <1>       jnc short loc_tcc_unlink_clusters
1437                              <1>
1438                              <1> pass_tcc_unlink_clusters:
1439 0000D5AF A2[AF940100]        <1>       mov   byte [TCC_FATErr], al
1440 0000D5B4 803D[F2910100]02    <1>       cmp   byte [FAT_BuffValidData], 2
1441 0000D5BB 750E                <1>       jne   short loc_tcc_calculate_FAT_freespace
1442 0000D5BD E89BFDFFFF          <1>       call  save_fat_buffer
1443 0000D5C2 7307                <1>       jnc   short loc_tcc_calculate_FAT_freespace
1444 0000D5C4 A2[AF940100]        <1>       mov   byte [TCC_FATErr], al ; Error
1445                              <1>       ;mov  byte [FAT_BuffValidData], 0
1446                              <1>
1447                              <1>       ; 01/03/2016
1448 0000D5C9 EB12                <1>       jmp   short loc_tcc_recalculate_FAT_freespace
1449                              <1>
1450                              <1> loc_tcc_calculate_FAT_freespace:
1451 0000D5CB A1[FA910100]        <1>       mov   eax, [FAT_ClusterCounter] ; signed (+-) number
1452 0000D5D0 66BB01FF            <1>       mov   bx, 0FF01h ; BH = FFh -> ESI = Dos drv desc. table
1453                              <1>                        ; BL = 1 -> add cluster
1454 0000D5D4 E819FEFFFF          <1>       call  calculate_fat_freespace
1455 0000D5D9 21C9                <1>       and   ecx, ecx ; cx = 0 -> valid free sector count
1456 0000D5DB 7409                <1>       jz    short pass_truncate_cc_recalc_FAT_freespace
1457                              <1>
1458                              <1> loc_tcc_recalculate_FAT_freespace:
1459 0000D5DD 66BB00FF            <1>       mov   bx, 0FF00h ; recalculate !
1460 0000D5E1 E80CFEFFFF          <1>       call  calculate_fat_freespace
1461                              <1>
1462                              <1> loc_tcc_calculate_FAT_freespace_err:
1463                              <1> pass_truncate_cc_recalc_FAT_freespace:
1464 0000D5E6 8B0D[FA910100]      <1>       mov   ecx, [FAT_ClusterCounter]
1465                              <1>
1466 0000D5EC 803D[AF940100]00    <1>       cmp   byte [TCC_FATErr], 0
1467 0000D5F3 7608                <1>       jna   short loc_tcc_unlink_clusters_retn
1468                              <1>
1469                              <1> loc_tcc_unlink_clusters_error:
1470 0000D5F5 0FB605[AF940100]    <1>       movzx  eax, byte [TCC_FATErr]
1471 0000D5FC F9                  <1>       stc
1472                              <1> loc_tcc_unlink_clusters_retn:
1473 0000D5FD C3                  <1>       retn
1474                              <1>
1475                              <1> set_fat32_fsinfo_sector_parms:
1476                              <1>       ; 15/10/2016
1477                              <1>       ; 23/03/2016
1478                              <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
1479                              <1>       ; INPUT ->
1480                              <1>       ;     ESI = Logical dos drive description table address
1481                              <1>       ;     [esi+LD_BPB+BPB_Reserved] = Free Cluster Count
1482                              <1>       ;     [esi+LD_BPB+BPB_Reserved+4] = First Free Cluster
1483                              <1>       ; OUTPUT ->
1484                              <1>       ;     ESI = Logical dos drive description table address
1485                              <1>       ;     CF = 0 -> OK..
1486                              <1>       ;     CF = 1 -> Error code in EAX (AL)
1487                              <1>       ;
1488                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
1489                              <1>
1490 0000D5FE E824000000          <1>       call  get_fat32_fsinfo_sector_parms
1491 0000D603 7221                <1>       jc    short update_fat32_fsinfo_sector_retn
1492                              <1>
1493 0000D605 8B463A              <1>       mov   eax, [esi+LD_BPB+BPB_Reserved] ; Free Cluster Count
1494 0000D608 8B563E              <1>       mov   edx, [esi+LD_BPB+BPB_Reserved+4] ; First free Cluster
1495                              <1>
1496                              <1>         ;mov ebx, DOSBootSectorBuff
1497 0000D60B 8983E8010000        <1>       mov   [ebx+488], eax
1498 0000D611 8993EC010000        <1>       mov   [ebx+492], edx
1499                              <1>
```

```
1500 0000D617 A1[9C940100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
1501 0000D61C B901000000          <1>       mov    ecx, 1
1502 0000D621 E86B550000          <1>       call   disk_write
1503                              <1>       ;jnc     short update_fat32_fsinfo_sector_retn
1504                              <1>
1505                              <1>       ; 15/10/2016 (1Dh -> 18)
1506                              <1>       ; 23/03/2016 (1Dh)
1507                              <1>       ;mov   eax, 18 ; Drive not ready or write error
1508                              <1>
1509                              <1> update_fat32_fsinfo_sector_retn:
1510 0000D626 C3                  <1>       retn
1511                              <1>
1512                              <1> get_fat32_fsinfo_sector_parms:
1513                              <1>       ; 15/10/2016
1514                              <1>       ; 23/03/2016
1515                              <1>       ; 01/03/2016
1516                              <1>       ; 29/02/2016 (TRDOS 386 =  TRDOS v2.0)
1517                              <1>       ; INPUT ->
1518                              <1>       ;      ESI = Logical dos drive description table address
1519                              <1>       ; OUTPUT ->
1520                              <1>       ;      ESI = Logical dos drive description table address
1521                              <1>       ;      EBX = FSINFO sector buffer address (DOSBootSectorBuff)
1522                              <1>       ;      CF = 0 -> OK..
1523                              <1>       ;         EAX = FsInfo sector address
1524                              <1>       ;         ECX = Free cluster count
1525                              <1>       ;         EDX = First free cluster
1526                              <1>       ;      CF = 1 -> Error code in AL (EAX)
1527                              <1>       ;         EBX = 0
1528                              <1>       ;
1529                              <1>       ;      [CFS_FAT32FSINFOSEC] = FAT32 FSINFO sector address
1530                              <1>         ;
1531                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
1532                              <1>
1533 0000D627 0FB74636            <1>       movzx  eax, word [esi+LD_BPB+FAT32_FSInfoSec]
1534 0000D62B 03466C              <1>       add    eax, [esi+LD_StartSector]
1535 0000D62E A3[9C940100]        <1>       mov    [CFS_FAT32FSINFOSEC], eax
1536                              <1>
1537 0000D633 BB[EE8F0100]        <1>         mov     ebx, DOSBootSectorBuff
1538 0000D638 B901000000          <1>       mov    ecx, 1
1539 0000D63D E85E550000          <1>       call   disk_read
1540 0000D642 7232                <1>       jc     short loc_read_FAT32_fsinfo_sec_err
1541                              <1>
1542 0000D644 BB[EE8F0100]        <1>       mov    ebx, DOSBootSectorBuff
1543                              <1>
1544 0000D649 813B52526141        <1>       cmp    dword [ebx], 41615252h
1545 0000D64F 751E                <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
1546                              <1>
1547 0000D651 81BBE4010000727241- <1>       cmp    dword [ebx+484], 61417272h
1547 0000D65A 61                  <1>
1548 0000D65B 7512                <1>       jne    short loc_read_FAT32_fsinfo_sec_stc
1549                              <1>
1550 0000D65D A1[9C940100]        <1>       mov    eax, [CFS_FAT32FSINFOSEC]
1551 0000D662 8B8BE8010000        <1>       mov    ecx, [ebx+488] ; free cluster count
1552 0000D668 8B93EC010000        <1>       mov    edx, [ebx+492] ; first (next) free cluster
1553                              <1>
1554 0000D66E C3                  <1>       retn
1555                              <1>
1556                              <1> loc_read_FAT32_fsinfo_sec_stc:
1557                              <1>       ; 15/10/2016 (0Bh -> 28)
1558 0000D66F B81C000000          <1>       mov    eax, 28 ; Invalid format!
1559 0000D674 EB05                <1>       jmp    short loc_read_FAT32_fsinfo_sec_stc_retn
1560                              <1>
1561                              <1> loc_read_FAT32_fsinfo_sec_err:
1562                              <1>       ; 15/10/2016 (15h -> 17)
1563                              <1>       ; 23/03/2016 (15h)
1564 0000D676 B811000000          <1>       mov    eax, 17 ; Drive not ready or read error
1565                              <1>
1566                              <1> loc_read_FAT32_fsinfo_sec_stc_retn:
1567 0000D67B 29DB                <1>       sub    ebx, ebx ; 0
1568 0000D67D F9                  <1>       stc
1569 0000D67E C3                  <1>       retn
1570                              <1>
1571                              <1> add_new_cluster:
1572                              <1>       ; 15/10/2016
1573                              <1>       ; 16/05/2016
1574                              <1>       ; 18/03/2016, 24/03/2016
1575                              <1>       ; 11/03/2016 (TRDOS 386 =  TRDOS v2.0)
1576                              <1>       ; 30/07/2011 (DRV_FAT.ASM)
1577                              <1>       ; 11/09/2010
1578                              <1>       ; INPUT ->
1579                              <1>       ;      ESI = Logical dos drv desc. table address
1580                              <1>       ;      EAX = Last cluster
1581                              <1>       ; OUTPUT ->
1582                              <1>       ;      ESI = Logical dos drv desc. table address
1583                              <1>       ;      EAX = New Last cluster (next cluster)
1584                              <1>       ;      cf = 1 -> error code in EAX (AL)
1585                              <1>       ;      cf = 1 -> DX = sectors per cluster
1586                              <1>       ;      ECX = Free sectors
1587                              <1>       ; NOTE:
1588                              <1>       ; This procedure does not update lm date&time !
1589                              <1>       ;
1590                              <1>       ; (Modified registers: EAX, EBX, ECX, EDX, EDI)
1591                              <1>       ;
1592                              <1>
1593 0000D67F A3[CC950100]        <1>       mov    [FAT_anc_LCluster], eax
1594                              <1>
1595 0000D684 E844F9FFFF          <1>       call   get_first_free_cluster
1596 0000D689 720B                <1>       jc     short loc_add_new_cluster_retn
1597                              <1>       ; EAX >= 2 and EAX < FFFFFFFFh is valid
1598                              <1>
1599 0000D68B 89C2                <1>       mov    edx, eax
1600                              <1>
1601 0000D68D 42                  <1>       inc    edx
1602                              <1>       ;jnz   short loc_add_new_cluster_check_ffc_eax
1603 0000D68E 7516                <1>       jnz    short loc_add_new_cluster_save_fcc
```

```
1604                                    <1>
1605                                    <1> loc_add_new_cluster_no_disk_space_retn:
1606 0000D690 B827000000                <1>         mov     eax, 27h ; MSDOS err => insufficient disk space
1607                                    <1> loc_add_new_cluster_stc_retn:
1608 0000D695 F9                        <1>         stc
1609                                    <1> loc_add_new_cluster_retn:
1610 0000D696 0FB65E13                  <1>         movzx   ebx, byte [esi+LD_BPB+SecPerClust]
1611 0000D69A 8B4E74                    <1>         mov     ecx, [esi+LD_FreeSectors]
1612                                    <1>         ;xor    edx, edx
1613                                    <1>         ;stc
1614 0000D69D C3                        <1>         retn
1615                                    <1>
1616                                    <1> loc_anc_invalid_format_stc_retn:
1617 0000D69E F9                        <1>         stc
1618                                    <1> loc_add_new_cluster_invalid_format_retn:
1619                                    <1>         ; 15/10/2016 (0Bh -> 28)
1620 0000D69F B81C000000                <1>         mov     eax, 28 ; Invalid format
1621 0000D6A4 EBF0                      <1>         jmp     short loc_add_new_cluster_retn
1622                                    <1>
1623                                    <1> ;loc_add_new_cluster_check_ffc_eax:
1624                                    <1> ;       cmp     eax, 2
1625                                    <1> ;       jb      short loc_add_new_cluster_invalid_format_retn
1626                                    <1>
1627                                    <1> loc_add_new_cluster_save_fcc:
1628 0000D6A6 A3[D0950100]              <1>         mov     [FAT_anc_FFCluster], eax
1629                                    <1>
1630 0000D6AB 83E802                    <1>         sub     eax, 2
1631 0000D6AE 0FB65E13                  <1>           movzx   ebx, byte [esi+LD_BPB+SecPerClust]
1632 0000D6B2 F7E3                      <1>         mul     ebx
1633 0000D6B4 09D2                      <1>         or      edx, edx
1634 0000D6B6 75E6                      <1>         jnz     short loc_anc_invalid_format_stc_retn
1635                                    <1>
1636                                    <1> loc_add_new_cluster_allocate_cluster:
1637                                    <1>         ; 18/03/2016
1638 0000D6B8 92                        <1>         xchg    edx, eax ; eax = 0
1639                                    <1>         ; 16/05/2016
1640                                    <1>         ;cmp    [ClusterBuffer_Valid], al ; 0
1641                                    <1>         ;jna    short loc_anc_clear_cluster_buffer
1642                                    <1>         ;; 'copy' command,
1643                                    <1>         ;; writing destination file clust after reading source file clust
1644                                    <1>         ;mov    [ClusterBuffer_Valid], al ; 0 ; reset
1645                                    <1>         ;jmp    short loc_add_new_cluster_write_nc_to_disk
1646                                    <1>
1647                                    <1> loc_anc_clear_cluster_buffer:
1648                                    <1>         ; 11/03/2016
1649                                    <1>         ; Clear buffer
1650 0000D6B9 BF00000700                <1>         mov     edi, Cluster_Buffer ; 70000h (for current TRDOS 386 version)
1651 0000D6BE 89D9                      <1>         mov     ecx, ebx ; sector count
1652 0000D6C0 C1E107                    <1>         shl     ecx, 7 ; 1 sector = 512 bytes -> 128 double words
1653                                    <1>         ;xor    eax, eax ; 0
1654 0000D6C3 F3AB                      <1>         rep     stosd
1655                                    <1>
1656                                    <1> loc_add_new_cluster_write_nc_to_disk:
1657                                    <1>         ; 11/03/2016
1658                                    <1>         ;xchg   eax, edx ; edx = 0, eax = sector offset
1659 0000D6C5 89D0                      <1>         mov     eax, edx
1660 0000D6C7 034668                    <1>           add     eax, [esi+LD_DATABegin]
1661 0000D6CA 72D3                      <1>         jc      short loc_add_new_cluster_invalid_format_retn
1662                                    <1>
1663 0000D6CC 89D9                      <1>         mov     ecx, ebx ; ECX = sectors per cluster (<256)
1664 0000D6CE BB00000700                <1>         mov     ebx, Cluster_Buffer
1665 0000D6D3 E8B9540000                <1>         call    disk_write
1666 0000D6D8 7307                      <1>         jnc     short loc_add_new_cluster_update_fat_nlc
1667                                    <1>
1668                                    <1>         ; 15/10/2016 (1Dh -> 18)
1669 0000D6DA B812000000                <1>         mov     eax, 18 ; Write Error
1670 0000D6DF EBB4                      <1>         jmp     short loc_add_new_cluster_stc_retn
1671                                    <1>
1672                                    <1> loc_add_new_cluster_update_fat_nlc:
1673 0000D6E1 A1[D0950100]              <1>         mov     eax, [FAT_anc_FFCluster]
1674 0000D6E6 31C9                      <1>         xor     ecx, ecx
1675 0000D6E8 890D[FA910100]            <1>         mov     [FAT_ClusterCounter], ecx ; 0 ; reset
1676 0000D6EE 49                        <1>         dec     ecx ; 0FFFFFFFFh
1677 0000D6EF E8ACF9FFFF                <1>         call    update_cluster
1678 0000D6F4 7304                      <1>         jnc     short loc_add_new_cluster_update_fat_plc
1679 0000D6F6 09C0                      <1>         or      eax, eax ;EAX = 0 -> cluster value is 0 or eocc
1680 0000D6F8 759B                      <1>         jnz     short loc_add_new_cluster_stc_retn
1681                                    <1>
1682                                    <1> loc_add_new_cluster_update_fat_plc:
1683 0000D6FA A1[CC950100]              <1>         mov     eax, [FAT_anc_LCluster]
1684 0000D6FF 8B0D[D0950100]            <1>         mov     ecx, [FAT_anc_FFCluster]
1685 0000D705 E896F9FFFF                <1>         call    update_cluster
1686 0000D70A 7314                      <1>         jnc     short loc_add_new_cluster_save_fat_buffer
1687 0000D70C 09C0                      <1>         or      eax, eax ; EAX = 0 -> cluster value is 0 or eocc
1688 0000D70E 7410                      <1>         jz      short loc_add_new_cluster_save_fat_buffer
1689                                    <1>
1690                                    <1> loc_anc_save_fat_buffer_err_retn:
1691                                    <1>         ;cmp    byte [FAT_ClusterCounter], 1
1692                                    <1>         ;jb     short loc_add_new_cluster_retn
1693                                    <1>
1694 0000D710 66BB00FF                  <1>         mov     bx, 0FF00h ; recalculate free space (BL = 0)
1695                                    <1>                            ; (BH = FFh -> Use ESI as Drv Param. Tbl.)
1696 0000D714 50                        <1>         push    eax
1697 0000D715 E8D8FCFFFF                <1>         call    calculate_fat_freespace
1698 0000D71A 58                        <1>         pop     eax
1699 0000D71B E975FFFFFF                <1>           jmp     loc_add_new_cluster_stc_retn
1700                                    <1>
1701                                    <1> loc_add_new_cluster_save_fat_buffer:
1702                                    <1>         ;cmp    byte [FAT_BuffValidData], 2
1703                                    <1>         ;jne    short loc_add_new_cluster_calc_FAT_freespace
1704                                    <1>         ;Byte [FAT_BuffValidData] =  2
1705 0000D720 E838FCFFFF                <1>         call    save_fat_buffer
1706 0000D725 72E9                      <1>         jc      short loc_anc_save_fat_buffer_err_retn
1707                                    <1>
1708                                    <1> loc_add_new_cluster_calc_FAT_freespace:
```

```
1709                                    <1>       ;mov   eax, 1 ; Only one Cluster
1710 0000D727 A1[FA910100]              <1>       mov    eax, [FAT_ClusterCounter]
1711 0000D72C 66BB01FF                  <1>       mov    bx, 0FF01h ; BH = FFh -> ESI -> Dos drv desc. table
1712                                    <1>              ; BL = 1 -> add cluster
1713 0000D730 B301                      <1>       mov    bl, 01h ; BL = 1 -> add clusters
1714                                    <1>       ; NOTE: EAX value will be added to Free Cluster Count
1715                                    <1>       ; (Free Cluster Count is decreased when EAX value is negative)
1716 0000D732 E8BBFCFFFF                <1>         call    calculate_fat_freespace
1717                                    <1>       ;ECX = 0 -> no error, ECX > 0 -> error or invalid return
1718 0000D737 21C9                      <1>       and    ecx, ecx ; ECX = 0 -> valid free sector count
1719 0000D739 7409                      <1>       jz     short loc_add_new_cluster_return_cluster_number
1720                                    <1>
1721                                    <1> loc_add_new_cluster_recalc_FAT_freespace:
1722 0000D73B 66BB00FF                  <1>       mov    bx, 0FF00h  ; recalculate free space
1723 0000D73F E8AEFCFFFF                <1>         call    calculate_fat_freespace
1724                                    <1>       ; cf = 0
1725                                    <1> loc_add_new_cluster_return_cluster_number:
1726 0000D744 89C1                      <1>       mov    ecx, eax ; Free sector count
1727 0000D746 A1[D0950100]              <1>       mov    eax, [FAT_anc_FFCluster]
1728 0000D74B 0FB65E13                  <1>       movzx  ebx, byte [esi+LD_BPB+SecPerClust]
1729                                    <1>       ;mov   edi, Cluster_Buffer
1730 0000D74F 31D2                      <1>       xor    edx, edx
1731 0000D751 C3                        <1>         retn
1732                                    <1>
1733                                    <1> write_cluster:
1734                                    <1>       ; 15/10/2016
1735                                    <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
1736                                    <1>       ;
1737                                    <1>       ; INPUT ->
1738                                    <1>       ;      EAX = Cluster Number (Sector index for SINGLIX FS)
1739                                    <1>       ;      ESI = Logical DOS Drive Description Table address
1740                                    <1>       ;      EBX = Cluster (File R/W) Buffer address (max. 64KB)
1741                                    <1>       ;      Only for SINGLIX FS:
1742                                    <1>       ;      EDX = File Number (The 1st FDT address)
1743                                    <1>       ; OUTPUT ->
1744                                    <1>       ;      cf = 1 -> Cluster can not be written onto disk
1745                                    <1>       ;          EAX > 0 -> Error number
1746                                    <1>       ;      cf = 0 -> Cluster has been written successfully
1747                                    <1>       ;
1748                                    <1>       ; (Modified registers: EAX, ECX, EBX, EDX)
1749                                    <1>
1750 0000D752 0FB64E13                  <1>       movzx  ecx, byte [esi+LD_BPB+BPB_SecPerClust]
1751                                    <1>       ; CL = 1 = [esi+LD_FS_Reserved2] ; SectPerClust for Singlix FS
1752                                    <1>
1753                                    <1> write_file_sectors: ; 16/03/2016
1754 0000D756 807E0300                  <1>       cmp    byte [esi+LD_FATType], 0
1755 0000D75A 761C                      <1>       jna    short write_fs_cluster
1756                                    <1>
1757                                    <1> write_fat_file_sectors:
1758 0000D75C 83E802                    <1>       sub    eax, 2 ; Beginning cluster number is always 2
1759 0000D75F 0FB65613                  <1>       movzx  edx, byte [esi+LD_BPB+BPB_SecPerClust] ; 18/03/2016
1760 0000D763 F7E2                      <1>       mul    edx
1761 0000D765 034668                    <1>       add    eax, [esi+LD_DATABegin] ; absolute address of the cluster
1762                                    <1>
1763                                    <1>       ; EAX = Disk sector address
1764                                    <1>       ; ECX = Sector count
1765                                    <1>       ; EBX = Buffer address
1766                                    <1>       ; (EDX = 0)
1767                                    <1>       ; ESI = Logical DOS drive description table address
1768                                    <1>
1769 0000D768 E824540000                <1>       call   disk_write
1770 0000D76D 7306                      <1>       jnc    short wclust_retn
1771                                    <1>
1772                                    <1>       ; 15/10/2016 (1Dh -> 18)
1773 0000D76F B812000000                <1>       mov    eax, 18 ; Drive not ready or write error !
1774 0000D774 C3                        <1>       retn
1775                                    <1>
1776                                    <1> wclust_retn:
1777 0000D775 29C0                      <1>       sub    eax, eax ; 0
1778 0000D777 C3                        <1>       retn
1779                                    <1>
1780                                    <1> write_fs_cluster:
1781                                    <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
1782                                    <1>       ; Singlix FS
1783                                    <1>
1784                                    <1>       ; EAX = Cluster number is sector index number of the file (eax)
1785                                    <1>
1786                                    <1>       ; EDX = File number is the first File Descriptor Table address
1787                                    <1>       ;      of the file. (Absolute address of the FDT).
1788                                    <1>
1789                                    <1>       ; eax = sector index (0 for the first sector)
1790                                    <1>       ; edx = FDT0 address
1791                                    <1>             ; 64 KB buffer = 128 sectors (limit)
1792 0000D778 B980000000                <1>       mov    ecx, 128 ; maximum count of sectors (before eof)
1793 0000D77D E801000000                <1>       call   write_fs_sectors
1794 0000D782 C3                        <1>       retn
1795                                    <1>
1796                                    <1> write_fs_sectors:
1797                                    <1>       ; 21/03/2016 (TRDOS 386 =  TRDOS v2.0)
1798 0000D783 F9                        <1>       stc
1799 0000D784 C3                        <1>       retn
1800                                    <1>
1801                                    <1> get_cluster_by_index:
1802                                    <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
1803                                    <1>       ; INPUT ->
1804                                    <1>       ;      EAX = Beginning cluster
1805                                    <1>       ;      EDX = Sector index in disk/file section
1806                                    <1>       ;            (Only for SINGLIX file system!)
1807                                    <1>       ;      ECX = Cluster sequence number after the beginning cluster
1808                                    <1>       ;      ESI = Logical DOS Drive Description Table address
1809                                    <1>       ; OUTPUT ->
1810                                    <1>       ;      EAX = Cluster number
1811                                    <1>       ;      cf = 1 -> Error code in AL (EAX)
1812                                    <1>       ;
1813                                    <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
```

```
1814                                    <1>       ;
1815 0000D785 807E0301                 <1>       cmp    byte [esi+LD_FATType], 1
1816 0000D789 721E                     <1>         jb     short get_fs_section_by_index
1817                                    <1>
1818 0000D78B 3B4E78                   <1>       cmp    ecx, [esi+LD_Clusters]
1819 0000D78E 7207                     <1>       jb     short gcbi_1
1820                                    <1> gcbi_0:
1821 0000D790 F9                       <1>       stc
1822 0000D791 B823000000               <1>       mov    eax, 23h ; Cluster not available !
1823                                    <1>                       ; MSDOS error code: FCB unavailable
1824 0000D796 C3                       <1>       retn
1825                                    <1> gcbi_1:
1826 0000D797 51                       <1>       push   ecx
1827 0000D798 E8D9F5FFFF               <1>       call   get_next_cluster
1828 0000D79D 59                       <1>       pop    ecx
1829 0000D79E 7203                     <1>       jc     short gcbi_3
1830 0000D7A0 E2F5                     <1>       loop   gcbi_1
1831                                    <1> gcbi_2:
1832 0000D7A2 C3                       <1>       retn
1833                                    <1> gcbi_3:
1834 0000D7A3 09C0                     <1>       or     eax, eax
1835 0000D7A5 74E9                     <1>       jz     short gcbi_0
1836 0000D7A7 F5                       <1>       cmc    ; stc
1837 0000D7A8 C3                       <1>       retn
1838                                    <1>
1839                                    <1> get_fs_section_by_index:
1840                                    <1>       ; 29/04/2016 (TRDOS 386 =  TRDOS v2.0)
1841                                    <1>       ; INPUT ->
1842                                    <1>       ;      EAX = Beginning FDT number/address
1843                                    <1>       ;      EDX = Sector index in disk/file section
1844                                    <1>       ;      ECX = Sector sequence number after the beginning FDT
1845                                    <1>       ;      ESI = Logical DOS Drive Description Table address
1846                                    <1>       ; OUTPUT ->
1847                                    <1>       ;      EAX = FDT number/address
1848                                    <1>       ;      EDX = Sector index of the section (0,1,2,3,4...)
1849                                    <1>       ;      cf = 1 -> Error code in AL (EAX)
1850                                    <1>       ;
1851                                    <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
1852                                    <1>       ;
1853 0000D7A9 B8FFFFFFFF               <1>       mov    eax, 0FFFFFFFFh
1854 0000D7AE C3                       <1>       retn
1855                                    <1>
1856                                    <1> get_last_section:
1857                                    <1>       ; 22/10/2016 (TRDOS 386 =  TRDOS v2.0)
1858                                    <1>       ; INPUT ->
1859                                    <1>       ;      EAX = (The 1st) FDT number/address
1860                                    <1>       ;      ESI = Logical DOS Drive Description Table address
1861                                    <1>       ; OUTPUT ->
1862                                    <1>       ;      EAX = FDT number/address of the last section
1863                                    <1>       ;      EDX = Last sector of the section (0,1,2,3,4...)
1864                                    <1>       ;      [glc_index] = sector index number of the last sector
1865                                    <1>       ;                    (for file, not for the last section)
1866                                    <1>       ;
1867                                    <1>       ;      cf = 1 -> Error code in AL (EAX)
1868                                    <1>       ;
1869                                    <1>       ;(Modified registers: EAX, ECX, EBX, EDX)
1870                                    <1>       ;
1871 0000D7AF B800000000               <1>       mov    eax, 0
1872 0000D7B4 BA00000000               <1>       mov    edx, 0
1873 0000D7B9 C3                       <1>       retn
3092                                        %include 'trdosk6.s' ; 24/01/2016
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.3) - MAIN PROGRAM : trdosk6.s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 02/03/2021
   5                                    <1> ; ----------------------------------------------------------------------------
   6                                    <1> ; Beginning: 24/01/2016
   7                                    <1> ; ----------------------------------------------------------------------------
   8                                    <1> ; Assembler: NASM version 2.15 t(trdos386.s)
   9                                    <1> ; ----------------------------------------------------------------------------
  10                                    <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
  11                                    <1> ; u1.s (27/17/2015), u2.s (03/01/2016)
  12                                    <1> ; ****************************************************************************
  13                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
  14                                    <1> ; TRDOS2.ASM (09/11/2011)
  15                                    <1> ; ----------------------------------------------------------------------------
  16                                    <1> ; INT_21H.ASM (c) 2009-2011 Erdogan TAN  [14/11/2009] Last Update: 08/11/2011
  17                                    <1>
  18                                    <1> sysent: ; < enter to system call >
  19                                    <1>       ; 17/03/2017
  20                                    <1>       ; 03/03/2017
  21                                    <1>       ; 19/02/2017
  22                                    <1>       ; 13/01/2017
  23                                    <1>       ; 06/06/2016
  24                                    <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
  25                                    <1>       ; 16/04/2015 - 19/10/2015 (Retro UNIX 386 v1)
  26                                    <1>       ; 10/04/2013 - 18/01/2014 (Retro UNIX 8086 v1)
  27                                    <1>       ;
  28                                    <1>       ; 'unkni' or 'sysent' is sytem entry from various traps.
  29                                    <1>       ; The trap type is determined and an indirect jump is made to
  30                                    <1>       ; the appropriate system call handler. If there is a trap inside
  31                                    <1>       ; the system a jump to panic is made. All user registers are saved
  32                                    <1>       ; and u.sp points to the end of the users stack. The sys (trap)
  33                                    <1>       ; instructor is decoded to get the the system code part (see
  34                                    <1>       ; trap instruction in the PDP-11 handbook) and from this
  35                                    <1>       ; the indirect jump address is calculated. If a bad system call is
  36                                    <1>       ; made, i.e., the limits of the jump table are exceeded, 'badsys'
  37                                    <1>       ; is called. If the call is legitimate control passes to the
  38                                    <1>       ; appropriate system routine.
  39                                    <1>       ;
  40                                    <1>       ; Calling sequence:
  41                                    <1>       ;     Through a trap caused by any sys call outside the system.
  42                                    <1>       ; Arguments:
  43                                    <1>       ;     Arguments of particular system call.
  44                                    <1>       ; ....................................................
```

```
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;       System call number is in EAX register.
                                <1>       ;
                                <1>       ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
                                <1>       ;     registers depending of function details.
                                <1>       ;
                                <1>       ; 16/04/2015
0000D7BA 368925[5C030300]       <1>         mov     [ss:u.sp], esp ; Kernel stack points to return address
                                <1>
                                <1>       ; save user registers
0000D7C1 1E                     <1>       push   ds
0000D7C2 06                     <1>       push   es
0000D7C3 0FA0                   <1>       push   fs
0000D7C5 0FA8                   <1>       push   gs
0000D7C7 60                     <1>       pushad  ; eax, ecx, edx, ebx, esp -before pushad-, ebp, esi, edi
                                <1>       ;
                                <1>       ; ESPACE = [ss:u.sp] - esp ; 4*12 = 48 ; 17/09/2015 ; 06/06/2016
                                <1>       ;      (ESPACE is size of space in kernel stack
                                <1>       ;      for saving/restoring user registers.)
                                <1>       ;
0000D7C8 50                     <1>       push   eax ; 01/07/2015
0000D7C9 66B81000               <1>       mov     ax, KDATA
0000D7CD 8ED8                   <1>         mov     ds, ax
0000D7CF 8EC0                   <1>         mov     es, ax
0000D7D1 8EE0                   <1>         mov     fs, ax
0000D7D3 8EE8                   <1>         mov     gs, ax
0000D7D5 A1[188A0100]           <1>       mov    eax, [k_page_dir]
0000D7DA 0F22D8                 <1>       mov    cr3, eax
0000D7DD 58                     <1>       pop    eax ; 01/07/2015
                                <1>       ; 19/10/2015
0000D7DE FC                     <1>       cld
                                <1>       ;
0000D7DF FE05[5B030300]         <1>       inc    byte [sysflg]
                                <1>              ; incb sysflg / indicate a system routine is in progress
0000D7E5 FB                     <1>         sti  ; 18/01/2014
0000D7E6 0F85DF9DFFFF           <1>       jnz     panic ; 24/05/2013
                                <1>              ; beq 1f
                                <1>              ; jmp panic ; / called if trap inside system
                                <1> ;1:
                                <1>       ; 17/03/2017
0000D7EC 80642438FE             <1>       and    byte [esp+ESPACE+8], ~1 ; clear carry flag
                                <1>
                                <1>       ; 16/04/2015
0000D7F1 A3[64030300]           <1>       mov    [u.r0], eax
0000D7F6 8925[60030300]         <1>       mov    [u.usp], esp ; kernel stack points to user's registers
                                <1>
                                <1>       ; 13/01/2017 (TRDOS 386 Feaure only !)
0000D7FC 803D[D4030300]00       <1>       cmp    byte [u.t_lock], 0 ; timer interrupt lock ?
0000D803 0F879D010000           <1>       ja     sysrele                ; yes, sys release only !!!
                                <1>
                                <1>              ; mov $s.syst+2,clockp
                                <1>              ; mov r0,-(sp) / save user registers
                                <1>              ; mov sp,u.r0 / pointer to bottom of users stack
                                <1>                       ; / in u.r0
                                <1>              ; mov r1,-(sp)
                                <1>              ; mov r2,-(sp)
                                <1>              ; mov r3,-(sp)
                                <1>              ; mov r4,-(sp)
                                <1>              ; mov r5,-(sp)
                                <1>              ; mov ac,-(sp) / "accumulator" register for extended
                                <1>                           ; / arithmetic unit
                                <1>              ; mov mq,-(sp) / "multiplier quotient" register for the
                                <1>                           ; / extended arithmetic unit
                                <1>              ; mov sc,-(sp) / "step count" register for the extended
                                <1>                           ; / arithmetic unit
                                <1>              ; mov sp,u.sp / u.sp points to top of users stack
                                <1>              ; mov 18.(sp),r0 / store pc in r0
                                <1>              ; mov -(r0),r0 / sys inst in r0      10400xxx
                                <1>              ; sub $sys,r0 / get xxx code
0000D809 C1E002                 <1>       shl    eax, 2
                                <1>              ; asl r0 / multiply by 2 to jump indirect in bytes
0000D80C 3DB8000000             <1>       cmp    eax, end_of_syscalls - syscalls
                                <1>              ; cmp r0,$2f-1f / limit of table (35) exceeded
                                <1>       ;jnb   short badsys
                                <1>              ; bhis badsys / yes, bad system call
0000D811 F5                     <1>       cmc
0000D812 9C                     <1>       pushf
0000D813 50                     <1>       push   eax
0000D814 8B2D[5C030300]         <1>       mov    ebp, [u.sp] ; Kernel stack at the beginning of sys call
0000D81A B0FE                   <1>       mov    al, 0FEh ; 11111110b
0000D81C 1400                   <1>       adc    al, 0 ; al = al + cf
0000D81E 204508                 <1>       and    [ebp+8], al ; flags (reset carry flag)
                                <1>              ; bic $341,20.(sp) / set users processor priority to 0
                                <1>                           ; / and clear carry bit
0000D821 5D                     <1>       pop    ebp ; eax
0000D822 9D                     <1>       popf
0000D823 0F8208020000           <1>         jc      badsys
0000D829 A1[64030300]           <1>       mov    eax, [u.r0]
                                <1>       ; system call registers: EAX, EDX, ECX, EBX, ESI, EDI
0000D82E FFA5[34D80000]         <1>       jmp    dword [ebp+syscalls]
                                <1>              ; jmp *1f(r0) / jump indirect thru table of addresses
                                <1>                          ; / to proper system routine.
                                <1> syscalls: ; 1:
                                <1>       ; 31/12/2017
                                <1>       ; 28/02/2017
                                <1>       ; 20/02/2017
                                <1>       ; 19/02/2017
                                <1>       ; 15/10/2016
                                <1>       ; 20/05/2016
                                <1>       ; 19/05/2016
                                <1>       ; 16/05/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 21/09/2015
                                <1>       ; 01/07/2015
```

```
                                <1>      ; 16/04/2015 (32 bit address modification)
0000D834 [8B1A0100]             <1>      dd sysver    ; 0 ; Get TRDOS 386 version number (v2.0)
0000D838 [93DA0000]             <1>      dd sysexit   ; 1
0000D83C [68DC0000]             <1>      dd sysfork   ; 2
0000D840 [9BE00000]             <1>      dd sysread   ; 3
0000D844 [BAE00000]             <1>      dd syswrite  ; 4
0000D848 [51DE0000]             <1>      dd sysopen   ; 5
0000D84C [72E00000]             <1>      dd sysclose  ; 6
0000D850 [EADB0000]             <1>      dd syswait   ; 7
0000D854 [80DD0000]             <1>      dd syscreat  ; 8
0000D858 [DF280100]             <1>      dd sysrename ; 9  ; TRDOS 386, Rename File (31/12/2017)
0000D85C [5A240100]             <1>      dd sysdelete ; 10 ; TRDOS 386, Delete File (29/12/2017)
0000D860 [280E0100]             <1>      dd sysexec   ; 11
0000D864 [84250100]             <1>      dd syschdir  ; 12
0000D868 [49270100]             <1>      dd systime   ; 13 ; TRDOS 386, Get Sys Date&Time (30/12/2017)
0000D86C [34E00000]             <1>      dd sysmkdir  ; 14
0000D870 [B8250100]             <1>      dd syschmod  ; 15 ; TRDOS 386, Change Attributes (30/12/2017)
0000D874 [C1240100]             <1>      dd sysrmdir  ; 16 ; TRDOS 386, Remove Directory (29/12/2017)
0000D878 [03110100]             <1>      dd sysbreak  ; 17
0000D87C [9E260100]             <1>      dd sysdrive  ; 18 ; TRDOS 386, Get/Set Current Drv (30/12/2017)
0000D880 [44110100]             <1>      dd sysseek   ; 19
0000D884 [56110100]             <1>      dd systell   ; 20
0000D888 [002A0100]             <1>      dd sysmem    ; 21 ; TRDOS 386, Get Total&Free Mem (31/12/2017)
0000D88C [362A0100]             <1>      dd sysprompt ; 22 ; TRDOS 386, Change Cmd Prompt (31/12/2017)
0000D890 [782A0100]             <1>      dd syspath   ; 23 ; TRDOS 386, Get/Set Run Path (31/12/2017)
0000D894 [E52A0100]             <1>      dd sysenv    ; 24 ; TRDOS 386, Get/Set Env Vars (31/12/2017)
0000D898 [CA270100]             <1>      dd sysstime  ; 25 ; TRDOS 386, Set Sys Date&Time (30/12/2017)
0000D89C [BC110100]             <1>      dd sysquit   ; 26
0000D8A0 [B0110100]             <1>      dd sysintr   ; 27
0000D8A4 [ED260100]             <1>      dd sysdir    ; 28 ; TRDOS 386, Get Curr Drive&Dir (30/12/2017)
0000D8A8 [51E10000]             <1>      dd sysemt    ; 29
0000D8AC [28270100]             <1>      dd sysldrvt  ; 30 ; TRDOS 386, Get Logical DOS DDT (30/12/2017)
0000D8B0 [02E30000]             <1>      dd sysvideo  ; 31 ; TRDOS 386 Video Functions (16/05/2016)
0000D8B4 [AF340100]             <1>      dd sysaudio  ; 32 ; TRDOS 386 Audio Functions (16/05/2016)
0000D8B8 [6AE10000]             <1>      dd systimer  ; 33 ; TRDOS 386 Timer Functions (18/05/2016)
0000D8BC [FD110100]             <1>      dd syssleep  ; 34 ; Retro UNIX 8086 v1 feature only !
                                <1>                         ; 11/06/2014
0000D8C0 [2C120100]             <1>      dd sysmsg    ; 35 ; Retro UNIX 386 v1 feature only !
                                <1>                         ; 01/07/2015
0000D8C4 [49130100]             <1>      dd sysgeterr ; 36 ; Retro UNIX 386 v1 feature only !
                                <1>                         ; 21/09/2015 - get last error number
0000D8C8 [31240100]             <1>      dd sysfpstat ; 37 ; TRDOS 386 FPU state option (28/02/2017)
0000D8CC [9A1A0100]             <1>      dd syspri    ; 38 ; change priority - TRDOS 386 (20/05/2016)
0000D8D0 [A6D90000]             <1>      dd sysrele   ; 39 ; TRDOS 386 (19/05/2016) (0 -> 39)
0000D8D4 [CD1B0100]             <1>      dd sysfff    ; 40 ; Find First File - TRDOS 386 (15/10/2016)
0000D8D8 [AC1C0100]             <1>      dd sysfnf    ; 41 ; Find Next File - TRDOS 386 (15/10/2016)
0000D8DC [1C230100]             <1>      dd sysalloc  ; 42 ; Allocate contiguous memory block/pages
                                <1>                         ; TRDOS 386 (19/02/2017) DMA buff fuctions
0000D8E0 [DA230100]             <1>      dd sysdalloc ; 43 ; Deallocate contiguous memory block/pages
                                <1>                         ; TRDOS 386 (19/02/2017) DMA buff fuctions
0000D8E4 [15240100]             <1>      dd syscalbac ; 44 ; IRQ Callback and Signal Response Byte
                                <1>                         ; service setup - TRDOS 386 (20/02/2017)
                                <1>                         ; 28/08/2017 (20/08/2017)
0000D8E8 [403D0100]             <1>      dd sysdma    ; 45 ; TRDOS 386 - (ISA) DMA service
                                <1>
                                <1> end_of_syscalls:
                                <1>
                                <1> error:
                                <1>      ; 18/05/2016
                                <1>      ; 13/05/2016
                                <1>      ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>      ; 16/04/2015 - 17/09/2015 (Retro UNIX 386 v1)
                                <1>      ; 10/04/2013 - 07/08/2013 (Retro UNIX 8086 v1)
                                <1>      ;
                                <1>      ; 'error' merely sets the error bit off the processor status (c-bit)
                                <1>      ; then falls right into the 'sysret', 'sysrele' return sequence.
                                <1>      ;
                                <1>      ; INPUTS -> none
                                <1>      ; OUTPUTS ->
                                <1>      ;     processor status - carry (c) bit is set (means error)
                                <1>      ;
                                <1>      ; 26/05/2013 (Stack pointer must be reset here!
                                <1>      ;           Because, jumps to error procedure
                                <1>      ;           disrupts push-pop nesting balance)
                                <1>      ;
0000D8EC 8B2D[5C030300]         <1>      mov    ebp, [u.sp] ; interrupt (system call) return (iretd) address
0000D8F2 804D0801               <1>      or     byte [ebp+8], 1  ; set carry bit of flags register
                                <1>                             ; (system call will return with cf = 1)
                                <1>             ; bis $1,20.(r1) / set c bit in processor status word below
                                <1>                              ; / users stack
                                <1>      ; 17/09/2015
0000D8F6 83ED30                 <1>      sub    ebp, ESPACE ; 48 ; total size of stack frame ('sysdefs.inc')
                                <1>                             ; for saving/restoring user registers
                                <1>      ;cmp   ebp, [u.usp]
                                <1>      ;je    short err0
0000D8F9 892D[60030300]         <1>      mov    [u.usp], ebp
                                <1> ;err0:
                                <1>      ; 01/09/2015
0000D8FF 8B25[60030300]         <1>      mov    esp, [u.usp]     ; Retro Unix 8086 v1 modification!
                                <1>                              ; 10/04/2013
                                <1>                              ; (If an I/O error occurs during disk I/O,
                                <1>                              ; related procedures will jump to 'error'
                                <1>                              ; procedure directly without returning to
                                <1>                              ; the caller procedure. So, stack pointer
                                <1>                                   ; must be restored here.)
                                <1>      ; 13/05/2016
                                <1>      ; NOTE: (The last) error code is in 'u.error', it can be retrieved by
                                <1>      ;     'get last error' system call later.
                                <1>
                                <1>      ; 03/09/2015 - 09/06/2015 - 07/08/2013
0000D905 C605[C6030300]00       <1>      mov    byte [u.kcall], 0 ; namei_r, mkdir_w reset
                                <1>
                                <1> sysret: ; < return from system call>
                                <1>      ; 01/03/2017
                                <1>      ; 28/02/2017
```

```
                        <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>       ; 16/04/2015 - 10/09/2015 (Retro UNIX 386 v1)
                        <1>       ; 10/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
                        <1>       ;
                        <1>       ; 'sysret' first checks to see if process is about to be
                        <1>       ; terminated (u.bsys). If it is, 'sysexit' is called.
                        <1>       ; If not, following happens:
                        <1>       ;     1) The user's stack pointer is restored.
                        <1>       ;     2) r1=0 and 'iget' is called to see if last mentioned
                        <1>       ;        i-node has been modified. If it has, it is written out
                        <1>       ;        via 'ppoke'.
                        <1>       ;     3) If the super block has been modified, it is written out
                        <1>       ;        via 'ppoke'.
                        <1>       ;     4) If the dismountable file system's super block has been
                        <1>       ;        modified, it is written out to the specified device
                        <1>       ;        via 'ppoke'.
                        <1>       ;     5) A check is made if user's time quantum (uquant) ran out
                        <1>       ;        during his execution. If so, 'tswap' is called to give
                        <1>       ;        another user a chance to run.
                        <1>       ;     6) 'sysret' now goes into 'sysrele'.
                        <1>       ;         (See 'sysrele' for conclusion.)
                        <1>       ;
                        <1>       ; Calling sequence:
                        <1>       ;     jump table or 'br sysret'
                        <1>       ; Arguments:
                        <1>       ;     -
                        <1>       ; ..........................................................
                        <1>       ;
                        <1>       ; ((AX=r1 for 'iget' input))
                        <1>       ;
0000D90C 31C0           <1>       xor   eax, eax ; 28/02/2017
                        <1> sysret0: ; 29/07/2015 (eax = 0, jump from sysexec)
0000D90E FEC0           <1>       inc   al ; 04/05/2013
0000D910 3805[B2030300] <1>       cmp   [u.bsys], al ; 1
                        <1>             ; tstb u.bsys / is a process about to be terminated because
0000D916 0F8377010000   <1>         jnb     sysexit ; 04/05/2013
                        <1>             ; bne sysexit / of an error? yes, go to sysexit
                        <1>       ;mov  esp, [u.usp] ; 24/05/2013 (that is not needed here)
                        <1>             ; mov u.sp,sp / no point stack to users stack
0000D91C FEC8           <1>       dec   al ; mov ax, 0
                        <1>             ; clr r1 / zero r1 to check last mentioned i-node
0000D91E E86D520000     <1>       call  iget
                        <1>             ; jsr r0,iget / if last mentioned i-node has been modified
                        <1>                           ; / it is written out
                        <1>       ; 10/01/2017
                        <1>       ; 09/01/2017
                        <1> ;sysrele: ; < release >
                        <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                        <1>       ; 16/04/2015 - 14/10/2015 (Retro UNIX 386 v1)
                        <1>       ; 10/04/2013 - 07/03/2014 (Retro UNIX 8086 v1)
                        <1>       ;
                        <1>       ; 'sysrele' first calls 'tswap' if the time quantum for a user is
                        <1>       ;  zero (see 'sysret'). It then restores the user's registers and
                        <1>       ; turns off the system flag. It then checked to see if there is
                        <1>       ; an interrupt from the user by calling 'isintr'. If there is,
                        <1>       ; the output gets flashed (see isintr) and interrupt action is
                        <1>       ; taken by a branch to 'intract'. If there is no interrupt from
                        <1>       ; the user, a rti is made.
                        <1>       ;
                        <1>       ; Calling sequence:
                        <1>       ;     Fall through a 'bne' in 'sysret' & ?
                        <1>       ; Arguments:
                        <1>       ;     -
                        <1>       ; ..........................................................
                        <1>       ;
                        <1>       ; 23/02/2014 (swapret)
                        <1>       ; 22/09/2013
                        <1> sysrel0: ;1:
0000D923 803D[A8030300]00 <1>       cmp   byte [u.quant], 0 ; 16/05/2013
                        <1>             ; tstb uquant / is the time quantum 0?
0000D92A 7705           <1>         ja      short swapret
                        <1>             ; bne 1f / no, don't swap it out
                        <1> sysrelease: ; 07/12/2013 (jump from 'clock')
0000D92C E822400000     <1>       call  tswap
                        <1>             ; jsr r0,tswap / yes, swap it out
                        <1>
                        <1> ; Retro Unix 8086 v1 feature: return from 'swap' to 'swapret' address.
                        <1> swapret: ;1:
                        <1>       ; 10/09/2015
                        <1>       ; 01/09/2015
                        <1>       ; 14/05/2015
                        <1>       ; 16/04/2015 (Retro UNIX 386 v1 - 32 bit, pm modifications)
                        <1>       ; 26/05/2013 (Retro UNIX 8086 v1)
                        <1>       ; cli
                        <1>       ; 24/07/2015
                        <1>       ;
                        <1>       ;; 'esp' must be already equal to '[u.usp]' here !
                        <1>       ;; mov esp, [u.usp]
                        <1>
                        <1>       ; 22/09/2013
0000D931 E85A520000     <1>       call  isintr
                        <1>       ; 20/10/2013
0000D936 7405           <1>       jz    short sysrel1
0000D938 E83F010000     <1>       call  intract
                        <1>             ; jsr r0,isintr / is there an interrupt from the user
                        <1>             ;     br intract / yes, output gets flushed, take interrupt
                        <1>                           ; / action
                        <1> sysrel1:
0000D93D FA             <1>       cli   ; 14/10/2015
                        <1> sysrel2:
                        <1>       ; 28/02/2017
                        <1>       ; Check if there is a (delayed) callback for current user/process
0000D93E A0[D7030300]   <1>       mov   al, [u.irqwait]
0000D943 240F           <1>       and   al, 0Fh ; is there a waiting IRQ callback service ?
0000D945 7444           <1>       jz    short sysrel8 ; no
```

```
                            <1>
                            <1>       ; Set return to IRQ callback service and return from the service
0000D947 0FB6D8             <1>       movzx  ebx, al
0000D94A 883D[D7030300]     <1>       mov    [u.irqwait], bh ; 0 ; reset
0000D950 8A9B[DA490100]     <1>       mov    bl, [ebx+IRQenum] ; (available) IRQ index +1 (1 to 9)
                            <1>       ; 01/03/2017
0000D956 FECB               <1>       dec    bl ; IRQ index number, 0 to 8
0000D958 7831               <1>       js     short sysrel8 ; 0 -> FFh (not in use!?)
                            <1>       ;
0000D95A A0[B3030300]       <1>       mov    al, [u.uno] ; current process (user) number
0000D95F 3883[4A9C0100]     <1>       cmp    [ebx+IRQ.owner], al
0000D965 7524               <1>       jne    short sysrel8 ; it is not the current user/process !?
0000D967 F683[5C9C0100]01   <1>       test   byte [ebx+IRQ.method], 1 ; callback ?
0000D96E 741B               <1>       jz     short sysrel8 ; not a callback method !?
                            <1>
0000D970 8B93[6E9C0100]     <1>       mov    edx, [ebx+IRQ.addr] ; IRQ callback service address (virtual)
0000D976 C605[D8030300]01   <1>       mov    byte [u.r_lock], 1 ; IRQ callback service in progress flag
                            <1>
0000D97D E879400000         <1>       call   wswap ; save user's registers & status
                            <1>              ;      (for return from IRQ callback service)
                            <1>
0000D982 8B2D[5C030300]     <1>       mov    ebp, [u.sp]; kernel's stack, points to EIP (user)
0000D988 895500             <1>       mov    [ebp], edx ; IRQ call back service address
                            <1> sysrel8:
0000D98B FE0D[5B030300]     <1>       dec    byte [sysflg]
                            <1>              ; decb sysflg / turn system flag off
                            <1>
0000D991 A1[B8030300]       <1>       mov    eax, [u.pgdir]
0000D996 0F22D8             <1>       mov    cr3, eax  ; 1st PDE points to Kernel Page Table 0 (1st 4 MB)
                            <1>                      ; (others are different than kernel page tables)
                            <1>       ; 10/09/2015
0000D999 61                 <1>       popad ; edi, esi, ebp, temp (icrement esp by 4), ebx, edx, ecx, eax
                            <1>             ; mov (sp)+,sc / restore user registers
                            <1>             ; mov (sp)+,mq
                            <1>             ; mov (sp)+,ac
                            <1>             ; mov (sp)+,r5
                            <1>             ; mov (sp)+,r4
                            <1>             ; mov (sp)+,r3
                            <1>             ; mov (sp)+,r2
                            <1>       ;
0000D99A A1[64030300]       <1>       mov    eax, [u.r0]  ; ((return value in EAX))
0000D99F 0FA9               <1>       pop    gs
0000D9A1 0FA1               <1>       pop    fs
0000D9A3 07                 <1>       pop    es
0000D9A4 1F                 <1>       pop    ds
                            <1>       ;or    word [esp+8], 200h ; 22/01/2017 ; force enabling interrupts
0000D9A5 CF                 <1>       iretd
                            <1>             ; rti / no, return from interrupt
                            <1>
                            <1> sysrele:
                            <1>       ; 24/03/2017
                            <1>       ; 28/02/2017
                            <1>       ; 27/02/2017
                            <1>       ; 29/01/2017
                            <1>       ; 14/01/2017
                            <1>       ; 13/01/2017
                            <1>       ; 09/01/2017, 10/01/2017, 12/01/2017
                            <1>       ; Major modification for TRDOS 386 (CallBack return)
                            <1>       ;
                            <1>       ; 'sysrele' system call restores previously saved
                            <1>       ; registers and addresses of the process
                            <1>       ; (Main purpose -in TRDOS 386- is to return from
                            <1>       ; timer callback service routine in ring 3 -user mode-.)
                            <1>       ;
                            <1>       ; check if the process is in timer callback phase
0000D9A6 803D[D4030300]00   <1>       cmp    byte [u.t_lock], 0 ; TIMER INT LOCK
                            <1>       ;je    short sysrel0 ; classic (Retro UNIX 386 type) sysrele
0000D9AD 7734               <1>       ja     short sysrel3
                            <1>       ; 27/02/2017
0000D9AF 803D[D8030300]00   <1>       cmp    byte [u.r_lock], 0 ; IRQ callback lock
0000D9B6 0F8667FFFFFF       <1>       jna    sysrel0 ; classic sysrele ; 24/03/2017
0000D9BC E859000000         <1>       call   sysrel7
0000D9C1 803D[D8030300]00   <1>       cmp    byte [u.r_lock], 0 ; IRQ callback service lock
0000D9C8 7628               <1>       jna    short sysrel4
0000D9CA C605[D8030300]00   <1>       mov    byte [u.r_lock], 0 ; reset
                            <1>       ;mov   byte [u.irqwait], 0 ; reset ; 28/02/2017
0000D9D1 A0[D9030300]       <1>       mov    al, [u.r_mode]
0000D9D6 08C0               <1>       or     al, al
0000D9D8 7518               <1>       jnz    short sysrel4
0000D9DA FEC8               <1>       dec    al
0000D9DC A2[D9030300]       <1>       mov    [u.r_mode], al ; 0FFh ; not necessary !?
0000D9E1 EB32               <1>       jmp    short sysrel6
                            <1> sysrel3:
                            <1>       ; 27/02/2017
0000D9E3 E832000000         <1>       call   sysrel7
                            <1>       ; 14/01/2017
0000D9E8 28C0               <1>       sub    al, al
0000D9EA 3805[D4030300]     <1>       cmp    [u.t_lock], al ; 0 ; TIMER INT LOCK
0000D9F0 770E               <1>       ja     short sysrel5 ; yes
                            <1> sysrel4:
                            <1>       ; 29/01/2017
0000D9F2 8B44241C           <1>       mov    eax, [esp+28] ; eax
0000D9F6 A3[64030300]       <1>       mov    [u.r0], eax
0000D9FB E93EFFFFFF         <1>       jmp    sysrel2
                            <1> sysrel5:
0000DA00 A2[D4030300]       <1>       mov    [u.t_lock], al ; 0 ; reset
0000DA05 A0[D5030300]       <1>       mov    al, [u.t_mode]
0000DA0A 20C0               <1>       and    al, al
                            <1>       ;jnz   short sysrel2 ; 0FFh ; user mode
0000DA0C 75E4               <1>       jnz    short sysrel4 ; 29/01/2017
0000DA0E FEC8               <1>       dec    al
0000DA10 A2[D5030300]       <1>       mov    [u.t_mode], al ; 0FFh ; not necessary !?
                            <1> sysrel6:
                            <1>       ; cpu will continue from the interrupted sytem call addr
0000DA15 61                 <1>       popad        ; edi, esi, ebp, esp, ebx, edx, ecx, eax
```

```
0000DA16 83C410             <1>       add    esp, 16      ; pass segment segisters: ds, es, fs, gs
0000DA19 CF                 <1>       iretd         ; eip, cs, eflags
                            <1>
                            <1> sysrel7:
0000DA1A 0FB61D[B3030300]   <1>       movzx  ebx, byte [u.uno] ; current process number
0000DA21 66C1E302           <1>       shl    bx, 2
                            <1>       ;cmp   [ebx+p.tcb-4], eax ; 0 ; is there callback address ?
                            <1>       ;jna   short sysrel0
                            <1>       ; yes, reset callback address then restore process registers
                            <1>       ;mov   [ebx+p.tcb-4], eax ; 0 ; reset
0000DA25 8B83[BC000300]     <1>       mov     eax, [ebx+p.upage-4] ; UPAGE address
0000DA2B FA                 <1>       cli    ; disable interrupts till 'iretd'
0000DA2C E902400000         <1>       jmp    rswap ; restore process 'u' structure
                            <1>
                            <1> badsys:
                            <1>       ; 25/12/2016
                            <1>       ; 18/04/2016 (TRDOS 386 = TRDOS v2.0)
                            <1>       ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
                            <1>       ; 03/02/2011 ('trdos_ifc_routine')
                            <1>       ;
                            <1>       ; 16/04/2015 (Retro UNIX 386 v1, 'badsys')
                            <1>       ; (EIP, EAX values will be shown on screen with error message)
                            <1>       ; (EIP = 'CD 40h' instruction address -INT 40h-)
                            <1>       ; (EAX = Function number)
                            <1>       ;
0000DA31 FE05[B2030300]     <1>       inc    byte [u.bsys]
                            <1>       ;
0000DA37 8B1D[5C030300]     <1>       mov    ebx, [u.sp] ; esp at the beginning of 'sysent'
0000DA3D 8B03               <1>       mov    eax, [ebx] ; EIP (return address, not 'INT 30h' address)
0000DA3F 83E802             <1>       sub    eax, 2 ; CDh, ##h
0000DA42 E8CE68FFFF         <1>       call   dwordtohex
0000DA47 8915[B3470100]     <1>       mov    [eip_str], edx
0000DA4D A3[B7470100]       <1>       mov    [eip_str+4], eax
0000DA52 A1[64030300]       <1>       mov    eax, [u.r0]
0000DA57 E8B968FFFF         <1>       call   dwordtohex
0000DA5C 8915[A2470100]     <1>       mov    [eax_str], edx
0000DA62 A3[A6470100]       <1>       mov    [eax_str+4], eax
                            <1>
0000DA67 66C705[97470100]34- <1>      mov    word [int_num_str], SYSCALL_INT_NUM ; 25/12/2016
0000DA6F 30                 <1>
                            <1>
0000DA70 BE[69470100]       <1>       mov    esi, ifc_msg ; "invalid funtion call !" msg (trdosk9.s)
0000DA75 E8EB9AFFFF         <1>       call   print_msg
                            <1>
0000DA7A EB17               <1>       jmp    sysexit
                            <1>
                            <1> intract: ; / interrupt action
                            <1>       ; 14/10/2015
                            <1>       ; 16/04/2015 (Retro UNIX 386 v1 - Beginning)
                            <1>       ; 09/05/2013 - 07/12/2013 (Retro UNIX 8086 v1)
                            <1>       ;
                            <1>       ; Retro UNIX 8086 v1 modification !
                            <1>       ; (Process/task switching and quit routine by using
                            <1>       ; Retro UNIX 8086 v1 keyboard interrupt output.))
                            <1>       ;
                            <1>       ; input -> 'u.quit'  (also value of 'u.intr' > 0)
                            <1>       ; output -> If value of 'u.quit' = FFFFh ('ctrl+brk' sign)
                            <1>       ;            'intract' will jump to 'sysexit'.
                            <1>       ;          Intract will return to the caller
                            <1>       ;            if value of 'u.quit' <> FFFFh.
                            <1>       ; 14/10/2015
0000DA7C FB                 <1>       sti
                            <1>       ; 07/12/2013
0000DA7D 66FF05[AC030300]   <1>       inc    word [u.quit]
0000DA84 7408               <1>       jz     short intrct0 ; FFFFh -> 0
0000DA86 66FF0D[AC030300]   <1>       dec    word [u.quit]
                            <1>       ; 16/04/2015
0000DA8D C3                 <1>       retn
                            <1> intrct0:
0000DA8E 58                 <1>       pop    eax ; call intract -> retn
                            <1>       ;
0000DA8F 31C0               <1>       xor    eax, eax
0000DA91 FEC0               <1>       inc    al  ; mov ax, 1
                            <1> ;;;
                            <1>       ; UNIX v1 original 'intract' routine...
                            <1>       ; / interrupt action
                            <1>             ;cmp *(sp),$rti / are you in a clock interrupt?
                            <1>             ; bne 1f / no, 1f
                            <1>             ; cmp (sp)+,(sp)+ / pop clock pointer
                            <1>       ; 1: / now in user area
                            <1>             ; mov r1,-(sp) / save r1
                            <1>             ; mov u.ttyp,r1
                            <1>                   ; / pointer to tty buffer in control-to r1
                            <1>             ; cmpb 6(r1),$177
                            <1>                   ; / is the interrupt char equal to "del"
                            <1>             ; beq 1f / yes, 1f
                            <1>             ; clrb 6(r1)
                            <1>                    ; / no, clear the byte
                            <1>                   ; / (must be a quit character)
                            <1>             ; mov (sp)+,r1 / restore r1
                            <1>             ; clr u.quit / clear quit flag
                            <1>             ; bis $20,2(sp)
                            <1>                   ; / set trace for quit (sets t bit of
                            <1>                   ; / ps-trace trap)
                            <1>             ; rti   ;  / return from interrupt
                            <1>       ; 1: / interrupt char = del
                            <1>             ; clrb 6(r1) / clear the interrupt byte
                            <1>                      ; / in the buffer
                            <1>             ; mov (sp)+,r1 / restore r1
                            <1>             ; cmp u.intr,$core / should control be
                            <1>                         ; / transferred to loc core?
                            <1>             ; blo 1f
                            <1>             ; jmp *u.intr / user to do rti yes,
                            <1>                         ; / transfer to loc core
                            <1>       ; 1:
```

```
                                <1>             ; sys 1 / exit
                                <1>
                                <1> sysexit: ; <terminate process>
                                <1>       ; 14/11/2017
                                <1>       ; 27/05/2017
                                <1>       ; 10/04/2017
                                <1>       ; 26/02/2017, 28/02/2017
                                <1>       ; 02/01/2017, 23/01/2017
                                <1>       ; 06/06/2016, 10/06/2016
                                <1>       ; 19/05/2016, 23/05/2016
                                <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; 16/04/2015 - 01/09/2015 (Retro UNIX 386 v1)
                                <1>       ; 19/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; 'sysexit' terminates a process. First each file that
                                <1>       ; the process has opened is closed by 'flose'. The process
                                <1>       ; status is then set to unused. The 'p.pid' table is then
                                <1>       ; searched to find children of the dying process. If any of
                                <1>       ; children are zombies (died by not waited for), they are
                                <1>       ; set free. The 'p.pid' table is then searched to find the
                                <1>       ; dying process's parent. When the parent is found, it is
                                <1>       ; checked to see if it is free or it is a zombie. If it is
                                <1>       ; one of these, the dying process just dies. If it is waiting
                                <1>       ; for a child process to die, it notified that it doesn't
                                <1>       ; have to wait anymore by setting it's status from 2 to 1
                                <1>       ; (waiting to active). It is awakened and put on runq by
                                <1>       ; 'putlu'. The dying process enters a zombie state in which
                                <1>       ; it will never be run again but stays around until a 'wait'
                                <1>       ; is completed by it's parent process. If the parent is not
                                <1>       ; found, process just dies. This means 'swap' is called with
                                <1>       ; 'u.uno=0'. What this does is the 'wswap' is not called
                                <1>       ; to write out the process and 'rswap' reads the new process
                                <1>       ; over the one that dies..i.e., the dying process is
                                <1>       ; overwritten and destroyed.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;     sysexit or conditional branch.
                                <1>       ; Arguments:
                                <1>       ;     -
                                <1>       ; ............................................................
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;       System call number (=1) is in EAX register.
                                <1>       ;
                                <1>       ;       Other parameters are in EDX, EBX, ECX, ESI, EDI, EBP
                                <1>       ;       registers depending of function details.
                                <1>       ;
                                <1>       ; ('swap' procedure is mostly different than original UNIX v1.)
                                <1>       ;
                                <1> ; / terminate process
                                <1>       ; AX = 1
0000DA93 6648                   <1>       dec    ax ; 0
0000DA95 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
0000DA9B FA                     <1>       cli    ; disable interrupts
                                <1>       ; 23/01/2017 - reset timer frequency (to 18.2Hz)
0000DA9C B036                   <1>       mov    al, 00110110b ; 36h
0000DA9E E643                   <1>       out    43h, al
0000DAA0 28C0                   <1>       sub    al, al ; 0
0000DAA2 E640                   <1>       out    40h, al ; LB
0000DAA4 E640                   <1>       out    40h, al ; HB
                                <1>       ;
0000DAA6 0FB61D[B3030300]       <1>       movzx  ebx, byte [u.uno]
                                <1>       ;mov   bl, [u.uno] ; process number of dying process
0000DAAD 3883[FF000300]         <1>       cmp    byte [ebx+p.timer-1], al ; 0
0000DAB3 763A                   <1>       jna    short sysexit_12 ; no timer events for this process
0000DAB5 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
0000DABB 8A0D[AB960100]         <1>       mov    cl, [timer_events] ; 14/11/2017
                                <1>       ;cli   ; disable interrupts
0000DAC1 B410                   <1>       mov    ah, 16 ; number of available timer events
0000DAC3 BE[60040300]           <1>       mov    esi, timer_set ; beginning address of timer events
                                <1> sysexit_7:
0000DAC8 8A06                   <1>       mov    al, [esi] ; process number (of timer event)
0000DACA 38D8                   <1>       cmp    al, bl ; process number comparison
0000DACC 7411                   <1>       je     short sysexit_10
0000DACE 20C0                   <1>       and    al, al
0000DAD0 7404                   <1>       jz     short sysexit_9
                                <1> sysexit_8:
0000DAD2 FEC9                   <1>       dec    cl
0000DAD4 7416                   <1>       jz     short sysexit_11
                                <1> sysexit_9:
0000DAD6 FECC                   <1>       dec    ah
0000DAD8 7415                   <1>       jz     short sysexit_12
0000DADA 83C610                 <1>       add    esi, 16
0000DADD EBE9                   <1>       jmp    short sysexit_7
                                <1>
                                <1> sysexit_10:
                                <1>       ;mov   byte [esi], 0
0000DADF 66C7060000             <1>       mov    word [esi], 0
```

```
                            <1>       ;mov   dword [esi+12], 0
                            <1>       ;
0000DAE4 FE0D[AB960100]     <1>       dec    byte [timer_events] ; 02/01/2017
                            <1>       ;
0000DAEA EBE6               <1>       jmp    short sysexit_8
                            <1>
                            <1> sysexit_11:
0000DAEC 6629C0             <1>       sub    ax, ax ; 0 ; 26/02/2017
                            <1> sysexit_12:
                            <1>       ; 26/02/2017 (Unlink IRQ callbacks belong to the user)
0000DAEF 803D[D6030300]00   <1>       cmp    byte [u.irqc], 0 ; Count of IRQ callbacks
0000DAF6 7E2E               <1>       jng    short sysexit_16 ; zero or invalid
                            <1>       ; 28/02/2017
                            <1>       ; clear IRQ callback flags (for 'sysrele' and 'sysret')
0000DAF8 A2[D7030300]       <1>       mov    [u.irqwait], al ; 0 ; force to clear waiting flag
0000DAFD A2[D8030300]       <1>       mov    [u.r_lock], al ; 0 ; force to clear busy flag
0000DB02 BE[4A9C0100]       <1>       mov    esi, IRQ.owner
                            <1> sysexit_13:
0000DB07 AC                 <1>       lodsb
0000DB08 3A05[B3030300]     <1>       cmp    al, [u.uno] ; owner = current user ?
0000DB0E 750C               <1>       jne    short sysexit_14
0000DB10 C646FF00           <1>       mov    byte [esi-1], 0 ; owner = 0 : Free
0000DB14 FE0D[D6030300]     <1>       dec    byte [u.irqc]
0000DB1A 7408               <1>       jz     short sysexit_15
                            <1> sysexit_14:
0000DB1C 81FE[529C0100]     <1>       cmp    esi, IRQ.owner + 8 ; the last IRQ index number ?
0000DB22 76E3               <1>       jna    short sysexit_13 ; no
                            <1> sysexit_15:
0000DB24 30C0               <1>       xor    al, al ; 0
                            <1> sysexit_16: ; 2:
0000DB26 FB                 <1>       sti    ; enable interrupts
                            <1>       ;
                            <1>       ; AX = 0
                            <1> sysexit_1: ; 1:
                            <1>       ; AX = File descriptor
                            <1>             ; / r1 has file descriptor (index to u.fp list)
                            <1>             ; / Search the whole list
0000DB27 E8F4340000         <1>       call   fclose
                            <1>             ; jsr r0,fclose / close all files the process opened
                            <1>       ;; ignore error return
                            <1>             ; br .+2 / ignore error return
                            <1>       ;inc   ax
0000DB2C FEC0               <1>       inc    al
                            <1>             ; inc r1 / increment file descriptor
                            <1>       ;cmp   ax, 10
0000DB2E 3C0A               <1>       cmp    al, 10
                            <1>             ; cmp r1,$10. / end of u.fp list?
0000DB30 72F5               <1>       jb     short sysexit_1
                            <1>             ; blt 1b / no, go back
                            <1>       ;movzx ebx, byte [u.uno]
0000DB32 8A1D[B3030300]     <1>       mov    bl, [u.uno] ; 02/01/2017
                            <1>             ; movb u.uno,r1 / yes, move dying process's number to r1
0000DB38 88A3[AF000300]     <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
                            <1>             ; clrb p.stat-1(r1) / free the process
                            <1>       ; 10/04/2017
0000DB3E 381D[C19C0100]     <1>       cmp    [audio_user], bl
0000DB44 7518               <1>       jne    short sysexit_17
                            <1>       ; reset audio device (current) owner and 'initializated' flag
0000DB46 883D[C19C0100]     <1>       mov    [audio_user], bh ; 0
                            <1>       ; 27/05/2017
0000DB4C 8B0D[AC9C0100]     <1>       mov    ecx,  [audio_buffer]
0000DB52 09C9               <1>       or     ecx, ecx
0000DB54 7408               <1>       jz     short sysexit_17
                            <1>       ; 'deallocate_user_pages' is not necessary in sysexit !!!
                            <1>       ;push  ebx
                            <1>       ;mov   ebx, ecx
                            <1>       ;mov   ecx, [audio_buff_size]
                            <1>       ;call  deallocate_user_pages
                            <1>       ;; (Modified Registers -> EAX, EDX, ESI, EDI, EBX, ECX, EBP)
0000DB56 29C9               <1>       sub    ecx, ecx
0000DB58 890D[AC9C0100]     <1>       mov    [audio_buffer], ecx ; 0
                            <1>       ;pop   ebx
                            <1> sysexit_17:
                            <1>       ;shl   bx, 1
0000DB5E D0E3               <1>       shl    bl, 1
                            <1>             ; asl r1 / use r1 for index into the below tables
0000DB60 668B8B[1E000300]   <1>       mov    cx, [ebx+p.pid-2]
                            <1>             ; mov p.pid-2(r1),r3 / move dying process's name to r3
0000DB67 668B93[3E000300]   <1>       mov    dx, [ebx+p.ppid-2]
                            <1>             ; mov p.ppid-2(r1),r4 / move its parents name to r4
                            <1>       ; xor  bx, bx ; 0
0000DB6E 30DB               <1>       xor    bl, bl ; 0
                            <1>             ; clr r2
0000DB70 31F6               <1>       xor    esi, esi ; 0
                            <1>             ; clr r5 / initialize reg
                            <1> sysexit_2: ; 1:
                            <1>              ; / find children of this dying process,
                            <1>             ; / if they are zombies, free them
                            <1>       ;add   bx, 2
0000DB72 80C302             <1>       add    bl, 2
                            <1>             ; add $2,r2 / search parent process table
                            <1>                       ; / for dying process's name
0000DB75 66398B[3E000300]   <1>       cmp    [ebx+p.ppid-2], cx
                            <1>             ; cmp p.ppid-2(r2),r3 / found it?
0000DB7C 7513               <1>       jne    short sysexit_4
                            <1>             ; bne 3f / no
                            <1>       ;shr   bx, 1
0000DB7E D0EB               <1>       shr    bl, 1
                            <1>             ; asr r2 / yes, it is a parent
0000DB80 80BB[AF000300]03   <1>       cmp    byte [ebx+p.stat-1], 3 ; SZOMB
                            <1>             ; cmpb p.stat-1(r2),$3 / is the child of this
                            <1>                                    ; / dying process a zombie
0000DB87 7506               <1>       jne    short sysexit_3
                            <1>             ; bne 2f / no
0000DB89 88A3[AF000300]     <1>       mov    [ebx+p.stat-1], ah ; 0, SFREE
```

```
                        <1>             ; clrb p.stat-1(r2) / yes, free the child process
                        <1> sysexit_3: ; 2:
                        <1>       ;shr  bx, 1
0000DB8F D0E3           <1>       shl   bl, 1
                        <1>             ; asl r2
                        <1> sysexit_4: ; 3:
                        <1>             ; / search the process name table
                        <1>             ; / for the dying process's parent
0000DB91 663993[1E000300]   <1>       cmp   [ebx+p.pid-2], dx
                        <1>             ; cmp p.pid-2(r2),r4 / found it?
0000DB98 7502           <1>       jne   short sysexit_5
                        <1>             ; bne 3f / no
0000DB9A 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
0000DB9C 80FB20         <1>       cmp   bl, nproc + nproc
                        <1>             ; cmp r2,$nproc+nproc / has whole table been searched?
0000DB9F 72D1           <1>       jb    short sysexit_2
                        <1>             ; blt 1b / no, go back
                        <1>             ; mov r5,r1 / yes, r1 now has parents process # x2
0000DBA1 21F6           <1>       and   esi, esi ; r5=r1
0000DBA3 7436           <1>       jz    short sysexit_6
                        <1>             ; beq 2f / no parent has been found.
                        <1>                   ; / The process just dies
0000DBA5 66D1EE         <1>       shr   si, 1
                        <1>             ; asr r1 / set up index to p.stat
0000DBA8 8A86[AF000300]     <1>       mov   al, [esi+p.stat-1]
                        <1>             ; movb p.stat-1(r1),r2 / move status of parent to r2
0000DBAE 20C0           <1>       and   al, al
0000DBB0 7429           <1>       jz    short sysexit_6
                        <1>             ; beq 2f / if its been freed, 2f
0000DBB2 3C03           <1>       cmp   al, 3
                        <1>             ; cmp r2,$3 / is parent a zombie?
0000DBB4 7425           <1>       je    short sysexit_6
                        <1>             ; beq 2f / yes, 2f
                        <1>       ; BH = 0
0000DBB6 8A1D[B3030300]     <1>       mov   bl, [u.uno]
                        <1>             ; movb u.uno,r3 / move dying process's number to r3
0000DBBC C683[AF000300]03   <1>       mov   byte [ebx+p.stat-1], 3  ; SZOMB
                        <1>             ; movb $3,p.stat-1(r3) / make the process a zombie
0000DBC3 3C01           <1>       cmp   al, 1 ; SRUN
0000DBC5 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
0000DBC7 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)
0000DBCE 6689F0         <1>       mov   ax, si ; r1  (process number in AL)
                        <1>       ;
                        <1>       ;mov  ebx, runq + 4
                        <1>             ; mov $runq+4,r2 / on the runq
0000DBD1 BB[54030300]       <1>       mov   ebx, runq+2 ; normal run queue ; 02/01/2017
0000DBD6 E8903E0000         <1>       call  putlu
                        <1>             ; jsr r0, putlu
                        <1> sysexit_6:
                        <1>             ; / the process dies
0000DBDB C605[B3030300]00   <1>       mov   byte [u.uno], 0
                        <1>             ; clrb u.uno / put zero as the process number,
                        <1>                 ; / so "swap" will
0000DBE2 E8863D0000         <1>       call  swap
                        <1>             ; jsr r0,swap / overwrite process with another process
                        <1> hlt_sys:
                        <1>       ;sti
                        <1> hlts0:
0000DBE7 F4             <1>       hlt
0000DBE8 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:
0000DBEA 0FB61D[B3030300]       <1>       movzx  ebx, byte [u.uno] ; 01/09/2015
                                <1>              ; movb u.uno,r1 / put parents process number in r1
0000DBF1 D0E3                   <1>       shl    bl, 1
                                <1>       ;shl   bx, 1
                                <1>              ; asl r1 / x2 to get index into p.pid table
0000DBF3 668B83[1E000300]       <1>       mov    ax, [ebx+p.pid-2]
                                <1>              ; mov p.pid-2(r1),r1 / get the name of this process
0000DBFA 31F6                   <1>       xor    esi, esi
                                <1>              ; clr r2
0000DBFC 31C9                   <1>       xor    ecx, ecx ; 30/10/2013
                                <1>       ;xor   cl, cl
                                <1>              ; clr r3 / initialize reg 3
                                <1> syswait_1: ; 1:
0000DBFE 6683C602               <1>       add    si, 2
                                <1>              ; add $2,r2 / use r2 for index into p.ppid table
                                <1>                      ; / search table of parent processes
                                <1>                      ; / for this process name
0000DC02 663B86[3E000300]       <1>       cmp    ax, [esi+p.ppid-2]
                                <1>              ; cmp p.ppid-2(r2),r1 / r2 will contain the childs
                                <1>                                    ; / process number
0000DC09 7535                   <1>       jne    short syswait_3
                                <1>              ;bne 3f / branch if no match of parent process name
                                <1>       ;inc   cx
0000DC0B FEC1                   <1>       inc    cl
                                <1>              ;inc r3 / yes, a match, r3 indicates number of children
0000DC0D 66D1EE                 <1>       shr    si, 1
                                <1>              ; asr r2 / r2/2 to get index to p.stat table
                                <1>       ; The possible states ('p.stat' values) of a process are:
                                <1>       ;      0 = free or unused
                                <1>       ;      1 = active
                                <1>       ;      2 = waiting for a child process to die
                                <1>       ;      3 = terminated, but not yet waited for (zombie).
0000DC10 80BE[AF000300]03       <1>       cmp    byte [esi+p.stat-1], 3 ; SZOMB, 05/02/2014
                                <1>              ; cmpb p.stat-1(r2),$3 / is the child process a zombie?
0000DC17 7524                   <1>       jne    short syswait_2
                                <1>              ; bne 2f / no, skip it
0000DC19 88BE[AF000300]         <1>       mov    [esi+p.stat-1], bh ; 0
                                <1>              ; clrb p.stat-1(r2) / yes, free it
0000DC1F 66D1E6                 <1>       shl    si, 1
                                <1>              ; asl r2 / r2x2 to get index into p.pid table
0000DC22 0FB786[1E000300]       <1>       movzx  eax, word [esi+p.pid-2]
0000DC29 A3[64030300]           <1>       mov    [u.r0], eax
                                <1>              ; mov p.pid-2(r2),*u.r0
                                <1>                           ; / put childs process name in (u.r0)
                                <1>       ;
                                <1>       ; Retro UNIX 386 v1 modification ! (17/09/2015)
                                <1>       ;
                                <1>       ; Parent process ID -p.ppid- field (of the child process)
                                <1>       ; must be cleared in order to prevent infinitive 'syswait'
                                <1>       ; system call loop from the application/program if it calls
                                <1>       ; 'syswait' again (mistakenly) while there is not a zombie
                                <1>       ; or running child process to wait. ('forktest.s', 17/09/2015)
                                <1>       ;
                                <1>       ; Note: syswait will return with error if there is not a
                                <1>       ;       zombie or running process to wait.
                                <1>       ;
0000DC2E 6629C0                 <1>       sub    ax, ax
0000DC31 668986[3E000300]       <1>       mov    [esi+p.ppid-2], ax ; 0 ; 17/09/2015
0000DC38 E9D1FCFFFF             <1>       jmp    sysret0 ; ax = 0
                                <1>       ;
                                <1>       ;jmp   sysret
                                <1>              ; br sysret1 / return cause child is dead
                                <1> syswait_2: ; 2:
0000DC3D 66D1E6                 <1>       shl    si, 1
                                <1>              ; asl r2 / r2x2 to get index into p.ppid table
                                <1> syswait_3: ; 3:
0000DC40 6683FE20               <1>       cmp    si, nproc+nproc
                                <1>              ; cmp r2,$nproc+nproc / have all processes been checked?
0000DC44 72B8                   <1>       jb     short syswait_1
                                <1>              ; blt 1b / no, continue search
                                <1>       ;and   cx, cx
0000DC46 20C9                   <1>       and    cl, cl
                                <1>              ; tst r3 / one gets here if there are no children
                                <1>                       ; / or children that are still active
                                <1>       ; 30/10/2013
0000DC48 750B                   <1>       jnz    short syswait_4
                                <1>       ;jz    error
                                <1>              ; beq error1 / there are no children, error
0000DC4A 890D[64030300]         <1>       mov    [u.r0], ecx ; 0
0000DC50 E997FCFFFF             <1>       jmp    error
                                <1> syswait_4:
0000DC55 8A1D[B3030300]         <1>       mov    bl, [u.uno]
                                <1>              ; movb u.uno,r1 / there are children so put
                                <1>                             ; / parent process number in r1
0000DC5B FE83[AF000300]         <1>       inc    byte [ebx+p.stat-1] ; 2, SWAIT, 05/02/2014
                                <1>              ; incb p.stat-1(r1) / it is waiting for
                                <1>                                  ; / other children to die
                                <1>       ; 04/11/2013
0000DC61 E8073D0000             <1>       call   swap
                                <1>              ; jsr r0,swap / swap it out, because it's waiting
0000DC66 EB82                   <1>       jmp    syswait_0
                                <1>              ; br syswait / wait on next process
                                <1>
                                <1> sysfork: ; < create a new process >
                                <1>       ; 02/01/2017 (TRDOS 386 modification)
                                <1>       ; 04/09/2015, 18/05/2015
```

```
; 28/08/2015, 01/09/2015, 02/09/2015
; 09/05/2015, 10/05/2015, 14/05/2015
; 06/05/2015 (Retro UNIX 386 v1 - Beginning)
; 24/05/2013 - 14/02/2014 (Retro UNIX 8086 v1)
;
; 'sysfork' creates a new process. This process is referred
; to as the child process. This new process core image is
; a copy of that of the caller of 'sysfork'. The only
; distinction is the return location and the fact that (u.r0)
; in the old process (parent) contains the process id (p.pid)
; of the new process (child). This id is used by 'syswait'.
; 'sysfork' works in the following manner:
;    1) The process status table (p.stat) is searched to find
;       a process number that is unused. If none are found
;       an error occurs.
;    2) when one is found, it becomes the child process number
;       and it's status (p.stat) is set to active.
;    3) If the parent had a control tty, the interrupt
;       character in that tty buffer is cleared.
;    4) The child process is put on the lowest priority run
;       queue via 'putlu'.
;    5) A new process name is gotten from 'mpid' (actually
;       it is a unique number) and is put in the child's unique
;       identifier; process id (p.pid).
;    6) The process name of the parent is then obtained and
;       placed in the unique identifier of the parent process
;       name is then put in 'u.r0'.
;    7) The child process is then written out on disk by
;       'wswap',i.e., the parent process is copied onto disk
;       and the child is born. (The child process is written
;       out on disk/drum with 'u.uno' being the child process
;       number.)
;    8) The parent process number is then restored to 'u.uno'.
;    9) The child process name is put in 'u.r0'.
;   10) The pc on the stack sp + 18 is incremented by 2 to
;       create the return address for the parent process.
;   11) The 'u.fp' list as then searched to see what files
;       the parent has opened. For each file the parent has
;       opened, the corresponding 'fsp' entry must be updated
;       to indicate that the child process also has opened
;       the file. A branch to 'sysret' is then made.

;
; Calling sequence:
;     from shell ?
; Arguments:
;     -
; Inputs: -
; Outputs: *u.r0 - child process name
; ..............................................................
;
; Retro UNIX 8086 v1 modification:
;     AX = r0 = PID (>0) (at the return of 'sysfork')
;     = process id of child a parent process returns
;     = process id of parent when a child process returns
;
;       In original UNIX v1, sysfork is called and returns as
;     in following manner: (with an example: c library, fork)
;
;     1:
;           sys   fork
;                 br 1f  / child process returns here
;           bes   2f     / parent process returns here
;           / pid of new process in r0
;           rts   pc
;     2: / parent process condionally branches here
;           mov   $-1,r0 / pid = -1 means error return
;           rts   pc
;
;     1: / child process brances here
;           clr   r0   / pid = 0 in child process
;           rts   pc
;
;     In UNIX v7x86 (386) by Robert Nordier (1999)
;           // pid = fork();
;           //
;           // pid == 0 in child process;
;           // pid == -1 means error return
;           // in child,
;           //     parents id is in par_uid if needed
;
;           _fork:
;                 mov   $.fork,eax
;                 int   $0x30
;                 jmp   1f
;                 jnc   2f
;                 jmp   cerror
;           1:
;                 mov   eax,_par_uid
;                 xor   eax,eax
;           2:
;                 ret
;
;     In Retro UNIX 8086 v1,
;     'sysfork' returns in following manner:
;
;           mov   ax, sys_fork
;           mov   bx, offset @f ; routine for child
;           int   20h
;           jc    error
;
;     ; Routine for parent process here (just after 'jc')
;           mov   word ptr [pid_of_child], ax
;           jmp   next_routine_for_parent
;
```

```
1093                                    <1>       ;      @@: ; routine for child process here
1094                                    <1>       ;           ....
1095                                    <1>       ;      NOTE: 'sysfork' returns to specified offset
1096                                    <1>       ;             for child process by using BX input.
1097                                    <1>       ;            (at first, parent process will return then
1098                                    <1>       ;            child process will return -after swapped in-
1099                                    <1>       ;            'syswait' is needed in parent process
1100                                    <1>       ;            if return from child process will be waited for.)
1101                                    <1>       ;
1102                                    <1>
1103                                    <1> ; / create a new process
1104                                    <1>       ; EBX = return address for child process
1105                                    <1>           ; (Retro UNIX 8086 v1 modification !)
1106 0000DC68 31F6                     <1>       xor    esi, esi
1107                                    <1>              ; clr r1
1108                                    <1> sysfork_1: ; 1: / search p.stat table for unused process number
1109 0000DC6A 46                       <1>       inc    esi
1110                                    <1>              ; inc r1
1111 0000DC6B 80BE[AF000300]00         <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE, 05/02/2014
1112                                    <1>              ; tstb p.stat-1(r1) / is process active, unused, dead
1113 0000DC72 760B                     <1>       jna    short sysfork_2
1114                                    <1>              ; beq 1f / it's unused so branch
1115 0000DC74 6683FE10                 <1>       cmp    si, nproc
1116                                    <1>              ; cmp r1,$nproc / all processes checked
1117 0000DC78 72F0                     <1>       jb     short sysfork_1
1118                                    <1>              ; blt 1b / no, branch back
1119                                    <1>       ;
1120                                    <1>       ; Retro UNIX 8086 v1. modification:
1121                                    <1>       ;      Parent process returns from 'sysfork' to address
1122                                    <1>       ;      which is just after 'sysfork' system call in parent
1123                                    <1>       ;      process. Child process returns to address which is put
1124                                    <1>       ;      in BX register by parent process for 'sysfork'.
1125                                    <1>       ;
1126                                    <1>              ;add $2,18.(sp) / add 2 to pc when trap occured, points
1127                                    <1>                              ; / to old process return
1128                                    <1>              ; br error1 / no room for a new process
1129 0000DC7A E96DFCFFFF               <1>       jmp    error
1130                                    <1> sysfork_2: ; 1:
1131 0000DC7F E8577FFFFF               <1>       call   allocate_page
1132 0000DC84 0F8262FCFFFF             <1>       jc     error
1133 0000DC8A 50                       <1>       push   eax   ; UPAGE (user structure page) address
1134                                    <1>       ; Retro UNIX 386 v1 modification!
1135 0000DC8B E85A81FFFF               <1>       call   duplicate_page_dir
1136                                    <1>              ; EAX = New page directory
1137 0000DC90 730B                     <1>       jnc    short sysfork_3
1138 0000DC92 58                       <1>       pop    eax   ; UPAGE (user structure page) address
1139 0000DC93 E82181FFFF               <1>       call   deallocate_page
1140 0000DC98 E94FFCFFFF               <1>       jmp    error
1141                                    <1> sysfork_3:
1142                                    <1>       ; Retro UNIX 386 v1 modification !
1143 0000DC9D 56                       <1>       push   esi
1144 0000DC9E E8583D0000               <1>       call   wswap ; save current user (u) structure, user registers
1145                                    <1>                    ; and interrupt return components (for IRET)
1146 0000DCA3 8705[B8030300]           <1>       xchg   eax, [u.pgdir] ; page directory of the child process
1147 0000DCA9 A3[BC030300]             <1>       mov    [u.ppgdir], eax ; page directory of the parent process
1148 0000DCAE 5E                       <1>       pop    esi
1149 0000DCAF 58                       <1>       pop    eax   ; UPAGE (user structure page) address
1150                                    <1>              ; [u.usp] = esp
1151 0000DCB0 89F7                     <1>       mov    edi, esi
1152 0000DCB2 66C1E702                 <1>       shl    di, 2
1153 0000DCB6 8987[BC000300]           <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
1154 0000DCBC A3[B4030300]             <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
1155                                    <1>       ; 28/08/2015
1156 0000DCC1 0FB605[B3030300]         <1>       movzx  eax, byte [u.uno] ; parent process number
1157                                    <1>              ; movb u.uno,-(sp) / save parent process number
1158 0000DCC8 89C7                     <1>       mov    edi, eax
1159 0000DCCA 50                       <1>         push eax ; **
1160 0000DCCB 8A87[7F000300]           <1>       mov     al, [edi+p.ttyc-1] ; console tty (parent)
1161                                    <1>       ; 18/09/2015
1162                                    <1>       ;mov     [esi+p.ttyc-1], al ; set child's console tty
1163                                    <1>       ;mov     [esi+p.waitc-1], ah ; 0 ; reset child's wait channel
1164 0000DCD1 668986[7F000300]         <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
1165                                    <1>                                ; ah - reset child's wait channel
1166 0000DCD8 89F0                     <1>       mov    eax, esi
1167 0000DCDA A2[B3030300]             <1>       mov    [u.uno], al ; child process number
1168                                    <1>              ;movb r1,u.uno / set child process number to r1
1169 0000DCDF FE86[AF000300]           <1>         inc     byte [esi+p.stat-1] ; 1, SRUN, 05/02/2014
1170                                    <1>              ; incb p.stat-1(r1) / set p.stat entry for child
1171                                    <1>                           ; / process to active status
1172                                    <1>              ; mov u.ttyp,r2 / put pointer to parent process'
1173                                    <1>                           ; / control tty buffer in r2
1174                                    <1>                 ; beq 2f / branch, if no such tty assigned
1175                                    <1>              ; clrb 6(r2) / clear interrupt character in tty buffer
1176                                    <1>       ; 2:
1177 0000DCE5 53                       <1>       push   ebx  ; * return address for the child process
1178                                    <1>                    ; * Retro UNIX 8086 v1 feature only !
1179                                    <1>       ; (Retro UNIX 8086 v1 modification!)
1180                                    <1>              ; mov $runq+4,r2
1181 0000DCE6 BB[54030300]             <1>       mov    ebx, runq+2 ; normal run queue ; 02/01/2017
1182 0000DCEB E87B3D0000               <1>       call   putlu
1183                                    <1>              ; jsr r0,putlu / put child process on lowest priority
1184                                    <1>                           ; / run queue
1185 0000DCF0 66D1E6                   <1>       shl    si, 1
1186                                    <1>              ; asl r1 / multiply r1 by 2 to get index
1187                                    <1>                     ; / into p.pid table
1188 0000DCF3 66FF05[4E030300]         <1>       inc    word [mpid]
1189                                    <1>              ; inc mpid / increment m.pid; get a new process name
1190 0000DCFA 66A1[4E030300]           <1>       mov    ax, [mpid]
1191 0000DD00 668986[1E000300]         <1>       mov    [esi+p.pid-2], ax
1192                                    <1>              ;mov mpid,p.pid-2(r1) / put new process name
1193                                    <1>                                   ; / in child process' name slot
1194 0000DD07 5A                       <1>       pop    edx  ; * return address for the child process
1195                                    <1>                    ; * Retro UNIX 8086 v1 feature only !
1196 0000DD08 5B                       <1>       pop    ebx  ; **
1197                                    <1>       ;mov   ebx, [esp] ; ** parent process number
```

```
1198                                    <1>             ; movb (sp),r2 / put parent process number in r2
1199 0000DD09 66D1E3                    <1>         shl    bx, 1
1200                                    <1>             ;asl r2 / multiply by 2 to get index into below tables
1201                                    <1>         ;movzx eax, word [ebx+p.pid-2]
1202 0000DD0C 668B83[1E000300]          <1>         mov    ax, [ebx+p.pid-2]
1203                                    <1>             ; mov p.pid-2(r2),r2 / get process name of parent
1204                                    <1>                                 ; / process
1205 0000DD13 668986[3E000300]          <1>         mov    [esi+p.ppid-2], ax
1206                                    <1>             ; mov r2,p.ppid-2(r1) / put parent process name
1207                                    <1>                              ; / in parent process slot for child
1208 0000DD1A A3[64030300]              <1>         mov    [u.r0], eax
1209                                    <1>             ; mov r2,*u.r0 / put parent process name on stack
1210                                    <1>                              ; / at location where r0 was saved
1211 0000DD1F 8B2D[5C030300]            <1>         mov    ebp, [u.sp] ; points to return address (EIP for IRET)
1212 0000DD25 895500                    <1>         mov    [ebp], edx ; *, CS:EIP -> EIP
1213                                    <1>                             ; * return address for the child process
1214                                    <1>             ; mov $sysret1,-(sp) /
1215                                    <1>             ; mov sp,u.usp / contents of sp at the time when
1216                                    <1>                              ; / user is swapped out
1217                                    <1>             ; mov $sstack,sp / point sp to swapping stack space
1218                                    <1>         ; 04/09/2015 - 01/09/2015
1219                                    <1>         ; [u.usp] = esp
1220 0000DD28 68[0CD90000]              <1>         push   sysret ; ***
1221 0000DD2D 8925[60030300]            <1>         mov    [u.usp], esp ; points to 'sysret' address (***)
1222                                    <1>                             ; (for child process)
1223 0000DD33 31C0                      <1>         xor    eax, eax
1224 0000DD35 66A3[94030300]            <1>         mov    [u.ttyp], ax ; 0
1225                                    <1>         ;
1226 0000DD3B E8BB3C0000                <1>         call   wswap ; Retro UNIX 8086 v1 modification !
1227                                    <1>             ;jsr r0,wswap / put child process out on drum
1228                                    <1>             ;jsr r0,unpack / unpack user stack
1229                                    <1>             ;mov u.usp,sp / restore user stack pointer
1230                                    <1>             ; tst (sp)+ / bump stack pointer
1231                                    <1>         ; Retro UNIX 386 v1 modification !
1232 0000DD40 58                        <1>         pop    eax ; ***
1233 0000DD41 66D1E3                    <1>         shl    bx, 1
1234 0000DD44 8B83[BC000300]            <1>         mov     eax, [ebx+p.upage-4] ; UPAGE address ; 14/05/2015
1235 0000DD4A E8E43C0000                <1>         call   rswap ; restore parent process 'u' structure,
1236                                    <1>                      ; registers and return address (for IRET)
1237                                    <1>             ;movb (sp)+,u.uno / put parent process number in u.uno
1238 0000DD4F 0FB705[4E030300]          <1>           movzx   eax, word [mpid]
1239 0000DD56 A3[64030300]              <1>         mov    [u.r0], eax
1240                                    <1>             ; mov mpid,*u.r0 / put child process name on stack
1241                                    <1>                              ; / where r0 was saved
1242                                    <1>             ; add $2,18.(sp) / add 2 to pc on stack; gives parent
1243                                    <1>                                ; / process return
1244                                    <1>         ;xor   ebx, ebx
1245 0000DD5B 31F6                      <1>         xor     esi, esi
1246                                    <1>             ;clr r1
1247                                    <1> sysfork_4: ; 1: / search u.fp list to find the files
1248                                    <1>                ; / opened by the parent process
1249                                    <1>         ; 01/09/2015
1250                                    <1>         ;xor   bh, bh
1251                                    <1>         ;mov   bl, [esi+u.fp]
1252 0000DD5D 8A86[6A030300]            <1>         mov    al, [esi+u.fp]
1253                                    <1>             ; movb u.fp(r1),r2 / get an open file for this process
1254                                    <1>           ;or      bl, bl
1255 0000DD63 08C0                      <1>         or     al, al
1256 0000DD65 740D                      <1>         jz     short sysfork_5
1257                                    <1>             ; beq 2f / file has not been opened by parent,
1258                                    <1>                    ; / so branch
1259 0000DD67 B40A                      <1>         mov    ah, 10 ; Retro UNIX 386 v1 fsp structure size = 10 bytes
1260 0000DD69 F6E4                      <1>         mul    ah
1261                                    <1>         ;movzx ebx, ax
1262 0000DD6B 6689C3                    <1>         mov    bx, ax
1263                                    <1>         ;shl     bx, 3
1264                                    <1>             ; asl r2 / multiply by 8
1265                                    <1>                   ; asl r2 / to get index into fsp table
1266                                    <1>                   ; asl r2
1267 0000DD6E FE83[4E010300]            <1>         inc     byte [ebx+fsp-2]
1268                                    <1>             ; incb fsp-2(r2) / increment number of processes
1269                                    <1>                          ; / using file, because child will now be
1270                                    <1>                          ; / using this file
1271                                    <1> sysfork_5: ; 2:
1272 0000DD74 46                        <1>           inc     esi
1273                                    <1>               ; inc r1 / get next open file
1274 0000DD75 6683FE0A                  <1>           cmp     si, 10
1275                                    <1>               ; cmp r1,$10. / 10. files is the maximum number which
1276                                    <1>                        ; / can be opened
1277 0000DD79 72E2                      <1>         jb     short sysfork_4
1278                                    <1>               ; blt 1b / check next entry
1279 0000DD7B E98CFBFFFF                <1>         jmp    sysret
1280                                    <1>               ; br sysret1
1281                                    <1>
1282                                    <1> syscreat: ; < create file >
1283                                    <1>         ; 13/11/2017
1284                                    <1>         ; 27/10/2016
1285                                    <1>         ; 25/10/2016, 26/10/2016
1286                                    <1>         ; 15/10/2016, 16/10/2016, 17/10/2016
1287                                    <1>         ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
1288                                    <1>         ;          -derived from INT_21H.ASM-
1289                                    <1>         ;            ("loc_INT21h_create_file")
1290                                    <1>           ;    10/07/2011 (12/03/2011)
1291                                    <1>           ;    INT 21h Function AH = 3Ch
1292                                    <1>           ;    Create File
1293                                    <1>           ;    INPUT
1294                                    <1>           ;       CX = Attributes
1295                                    <1>           ;          DS:DX= Address of zero terminaned path name
1296                                    <1>           ;
1297                                    <1>         ; 27/12/2015 (Retro UNIX 386 v1.1)
1298                                    <1>         ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
1299                                    <1>         ; 27/05/2013 (Retro UNIX 8086 v1)
1300                                    <1>         ;
1301                                    <1>         ; 'syscreat' called with two arguments; name and mode.
1302                                    <1>         ; u.namep points to name of the file and mode is put
```

```
1303                                  <1>       ; on the stack. 'namei' is called to get i-number of the file.
1304                                  <1>       ; If the file aready exists, it's mode and owner remain
1305                                  <1>       ; unchanged, but it is truncated to zero length. If the file
1306                                  <1>       ; did not exist, an i-node is created with the new mode via
1307                                  <1>       ; 'maknod' whether or not the file already existed, it is
1308                                  <1>       ; open for writing. The fsp table is then searched for a free
1309                                  <1>       ; entry. When a free entry is found, proper data is placed
1310                                  <1>       ; in it and the number of this entry is put in the u.fp list.
1311                                  <1>       ; The index to the u.fp (also know as the file descriptor)
1312                                  <1>       ; is put in the user's r0.
1313                                  <1>       ;
1314                                  <1>       ; Calling sequence:
1315                                  <1>       ;     syscreate; name; mode
1316                                  <1>       ; Arguments:
1317                                  <1>       ;     name - name of the file to be created
1318                                  <1>       ;     mode - mode of the file to be created
1319                                  <1>       ; Inputs: (arguments)
1320                                  <1>       ; Outputs: *u.r0 - index to u.fp list
1321                                  <1>       ;               (the file descriptor of new file)
1322                                  <1>       ; ..........................................................
1323                                  <1>       ;
1324                                  <1>       ; Retro UNIX 8086 v1 modification:
1325                                  <1>       ;      'syscreate' system call has two arguments; so,
1326                                  <1>       ;     * 1st argument, name is pointed to by BX register
1327                                  <1>       ;     * 2nd argument, mode is in CX register
1328                                  <1>       ;
1329                                  <1>       ;     AX register (will be restored via 'u.r0') will return
1330                                  <1>       ;     to the user with the file descriptor/number
1331                                  <1>       ;     (index to u.fp list).
1332                                  <1>       ;
1333                                  <1>       ;call  arg2
1334                                  <1>       ; * name - 'u.namep' points to address of file/path name
1335                                  <1>       ;          in the user's program segment ('u.segmnt')
1336                                  <1>       ;          with offset in BX register (as sysopen argument 1).
1337                                  <1>       ; * mode - sysopen argument 2 is in CX register
1338                                  <1>       ;          which is on top of stack.
1339                                  <1>       ;
1340                                  <1>       ; TRDOS 386 (10/10/2016)
1341                                  <1>       ;
1342                                  <1>         ; INPUT ->
1343                                  <1>         ;      CL = File Attributes
1344                                  <1>       ;           bit 0 (1) - Read only file (R)
1345                                  <1>       ;            bit 1 (1) - Hidden file (H)
1346                                  <1>         ;             bit 2 (1) - System file (R)
1347                                  <1>       ;            bit 3 (1) - Volume label/name (V)
1348                                  <1>         ;             bit 4 (1) - Subdirectory (D)
1349                                  <1>       ;            bit 5 (1) - File has been archived (A)
1350                                  <1>         ;          EBX = Pointer to filename (ASCIIZ) -path-
1351                                  <1>       ;
1352                                  <1>       ; OUTPUT ->
1353                                  <1>       ;          eax = File/Device Handle/Number (index) (AL)
1354                                  <1>       ;          cf = 1 -> Error code in AL
1355                                  <1>       ;
1356                                  <1>       ; Modified Registers: EAX (at the return of system call)
1357                                  <1>       ;
1358                                  <1>       ; Note: If the file is existing and it has not any one
1359                                  <1>       ;     of S,H,R,V,D attributes, it will be truncated
1360                                  <1>       ;     to zero length; otherwise, access error will be
1361                                  <1>       ;     returned.
1362                                  <1>
1363                                  <1> sysmkdir_0:
1364 0000DD80 F6C108                 <1>       test   cl, 08h ; Volume name
1365 0000DD83 740A                   <1>       jz     short syscreat_0
1366                                  <1>
1367                                  <1>       ; Volume name or long name creation
1368                                  <1>       ; is not permitted (in TRDOS 386)!
1369 0000DD85 B80B000000             <1>       mov    eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
1370 0000DD8A E926020000             <1>         jmp  sysopen_dev_err
1371                                  <1>
1372                                  <1> syscreat_0:
1373                                  <1>         ;mov [u.namep], ebx
1374 0000DD8F 51                     <1>       push   ecx
1375 0000DD90 89DE                   <1>       mov    esi, ebx
1376                                  <1>       ; file name is forced, change directory as temporary
1377                                  <1>       ;mov   ax, 1
1378                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
1379                                  <1>       ;call  set_working_path
1380 0000DD92 E82C520000             <1>       call   set_working_path_x ; 17/10/2016
1381 0000DD97 0F82D7000000           <1>       jc     syscreat_err
1382                                  <1>
1383                                  <1>       ; 16/10/2016
1384 0000DD9D 803D[CF960100]00       <1>       cmp    byte [SWP_inv_fname], 0
1385 0000DDA4 776C                   <1>       ja     short  syscreat_inv_fname ; invalid file name !
1386                                  <1>
1387                                  <1>       ; Here, we have a valid path and also a valid file name
1388                                  <1>       ; (Working dir has been changed if the path
1389                                  <1>       ;  -file name string- had contained a dir name.)
1390                                  <1>
1391 0000DDA6 6631C0                 <1>       xor    ax, ax
1392                                  <1>       ;mov   esi, FindFile_Name
1393 0000DDA9 E8E3B6FFFF             <1>       call   find_first_file
1394 0000DDAE 59                     <1>       pop    ecx
1395                                  <1>              ; ESI = Directory Entry (FindFile_DirEntry) Location
1396                                  <1>              ; EDI = Directory Buffer Directory Entry Location
1397                                  <1>              ; EAX = File Size
1398                                  <1>              ;  BL = Attributes of The File/Directory
1399                                  <1>              ;  BH = Long Name Yes/No Status (>0 is YES)
1400                                  <1>              ;  DX > 0 : Ambiguous filename chars are used
1401 0000DDAF 7269                   <1>       jc     short syscreat_1 ; file not found (the good!)
1402                                  <1>                             ; or another error (the bad')
1403                                  <1>
1404                                  <1>       ; (& the uggly!) truncate file to zero length before open
1405                                  <1>
1406                                  <1>       ;'*' and '?' already checked at 'set_working_path' stage
1407                                  <1>       ;and   dx, dx
```

```
1408                                    <1>       ;jnz   short sysmkdir_err ; permission denied
1409                                    <1>                               ; invalid filename chars
1410                                    <1>
1411                                    <1>       ;test  cl, 10h ; subdirectory ?
1412                                    <1>       ;jnz   short sysmkdir_err
1413                                    <1>
1414                                    <1>       ; BL = File Attributes:
1415                                    <1>       ;            bit 0 (1) - Read only file (R)
1416                                    <1>       ;              bit 1 (1) - Hidden file (H)
1417                                    <1>         ;              bit 2 (1) - System file (R)
1418                                    <1>       ;              bit 3 (1) - Volume label/name (V)
1419                                    <1>         ;              bit 4 (1) - Subdirectory (D)
1420                                    <1>       ;            bit 5 (1) - File has been archived
1421                                    <1>
1422                                    <1>       ; * existing directory must not be truncated
1423                                    <1>       ;   (we don't know it is empty or not, at this stage)
1424                                    <1>       ; * existing volume name (or a long name) can not be
1425                                    <1>       ;   re-created or truncated by 'syscreat'
1426                                    <1>       ; * A file with S, H, R attributes must not be truncated
1427                                    <1>       ;   (change attributes to normal, if you need truncate it)
1428                                    <1>
1429 0000DDB1 F6C31F                   <1>       test   bl, 00011111b  ; check attributes of existing file
1430 0000DDB4 754E                     <1>       jnz    short sysmkdir_err
1431                                    <1>
1432                                    <1>       ;; normal file, OK to continue...
1433                                    <1>
1434                                    <1>       ; ESI = FindFile_DirEntry
1435 0000DDB6 668B4614                 <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
1436 0000DDBA C1E010                   <1>       shl    eax, 16 ; 13/11/2017
1437 0000DDBD 668B461A                 <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
1438                                    <1>       ; EAX = First cluster to be truncated/unlinked
1439 0000DDC1 57                       <1>       push   edi
1440 0000DDC2 51                       <1>       push   ecx
1441 0000DDC3 BE00010900               <1>       mov    esi, Logical_DOSDisks
1442 0000DDC8 29C9                     <1>       sub    ecx, ecx
1443 0000DDCA 8A2D[DE8A0100]           <1>       mov    ch, [Current_Drv]
1444 0000DDD0 01CE                     <1>       add    esi, ecx
1445                                    <1>       ; ESI = Logical dos drive description table address
1446 0000DDD2 E8C9F7FFFF               <1>       call   truncate_cluster_chain
1447 0000DDD7 59                       <1>       pop    ecx
1448 0000DDD8 5F                       <1>       pop    edi
1449 0000DDD9 7230                     <1>       jc     short syscreate_truncate_err
1450                                    <1>
1451                                    <1>       ; 26/10/2016
1452                                    <1>       ; EDI = Directory entry address in directory buffer
1453                                    <1>       ; Update directory entry
1454 0000DDDB E848DCFFFF               <1>       call   convert_current_date_time
1455                                    <1>       ; OUTPUT -> DX = Date in dos dir entry format
1456                                    <1>         ;        AX = Time in dos dir entry format
1457 0000DDE0 66894716                 <1>       mov    [edi+DirEntry_WrtTime], ax
1458 0000DDE4 66895718                 <1>       mov    [edi+DirEntry_WrtDate], dx
1459 0000DDE8 66895712                 <1>       mov    [edi+DirEntry_LastAccDate], dx
1460 0000DDEC 31C0                     <1>       xor    eax, eax ; file size = 0
1461 0000DDEE 89471C                   <1>       mov    [edi+DirEntry_FileSize], eax ; 0
1462 0000DDF1 C605[04920100]02         <1>       mov    byte [DirBuff_ValidData], 2 ; data changed sign
1463 0000DDF8 BE[D0930100]             <1>       mov    esi, FindFile_DirEntry
1464 0000DDFD B201                     <1>       mov    dl, 1 ; open file for writing
1465 0000DDFF E9AA000000               <1>       jmp    sysopen_2
1466                                    <1>
1467                                    <1> sysmkdir_err:
1468                                    <1>       ; 1 = write, 2 = read & write, >2 = invalid
1469 0000DE04 B80B000000               <1>         mov  eax, ERR_FILE_ACCESS  ; 11 ; 'permission denied !'
1470 0000DE09 EB73                     <1>         jmp  short sysopen_err
1471                                    <1>
1472                                    <1> syscreate_truncate_err:
1473 0000DE0B B812000000               <1>       mov    eax, ERR_DRV_WRITE ; 18 ; 'disk write error !'
1474 0000DE10 EB6C                     <1>         jmp  short sysopen_err
1475                                    <1>
1476                                    <1> syscreat_inv_fname:  ; invalid file name chars
1477                                    <1>       ; 16/10/2016
1478 0000DE12 B81A000000               <1>       mov    eax, ERR_INV_FILE_NAME  ; 26 ; invalid file name chars
1479 0000DE17 59                       <1>       pop    ecx
1480 0000DE18 EB64                     <1>       jmp    sysopen_err
1481                                    <1>
1482                                    <1> syscreat_1:
1483                                    <1>       ; Error code in EAX
1484 0000DE1A 3C02                     <1>         cmp  al, 02h ; 'File not found' error
1485 0000DE1C 7560                     <1>         jne  sysopen_err
1486                                    <1>
1487 0000DE1E F6C110                   <1>       test   cl, 10h ; Directory
1488 0000DE21 0F852C020000             <1>       jnz    sysmkdir_2
1489                                    <1>
1490                                    <1> syscreat_2:
1491 0000DE27 BE[C0930100]             <1>       mov    esi, FindFile_Name
1492                                    <1>         ;xor edx, edx
1493 0000DE2C 31C0                     <1>         xor  eax, eax ; File Size  = 0
1494 0000DE2E 31DB                     <1>       xor    ebx, ebx
1495 0000DE30 4B                       <1>       dec    ebx ; FFFFFFFFh -> create empty file
1496                                    <1>                  ;              (only for FAT fs)
1497                                    <1>       ; CL = File Attributes
1498 0000DE31 E8F8EBFFFF               <1>       call   create_file
1499 0000DE36 7246                     <1>       jc     sysopen_err
1500                                    <1>              ; EAX = New file's first cluster
1501                                    <1>              ; ESI = Logical Dos Drv Descr. Table Addr.
1502                                    <1>              ; EBX = offset CreateFile_Size
1503                                    <1>              ; ECX = Sectors per cluster (<256)
1504                                    <1>              ; EDX = Directory entry index/number (<65536)
1505                                    <1>       ; 26/10/2016
1506                                    <1>       ;mov   esi, Directory_Buffer
1507                                    <1>       ;shl   dx, 5 ; *32
1508                                    <1>       ;add   esi, edx
1509                                    <1>       ;; esi = directory entry address in directory buffer
1510                                    <1>
1511                                    <1>       ; Here, directory entry has been created but last
1512                                    <1>       ; modification date & time of the parent dir has not
```

```
                                <1>       ; been updated, yet!
                                <1>       ; (Note: Directory and FAT buffers have been updated...)
                                <1>
0000DE38 E824DDFFFF             <1>       call   update_parent_dir_lmdt ; now, it is OK too!
                                <1>
                                <1>       ; 25/10/2016
0000DE3D 66B80018               <1>       mov    ax, 1800h
0000DE41 BE[C0930100]           <1>       mov    esi, FindFile_Name
0000DE46 E846B6FFFF             <1>       call   find_first_file
0000DE4B 7231                   <1>       jc     short sysopen_err
                                <1>
                                <1>       ; Only possible error after here is
                                <1>       ; "too many open files !" error.
                                <1>       ;
                                <1>       ; If "syscreat" will return with that error,
                                <1>       ; (the file has been created but it could not be opened)
                                <1>       ; the user must retry to open this file again
                                <1>       ; or must close another file before using
                                <1>       ; "sysopen" system call.
                                <1>
0000DE4D B201                   <1>       mov    dl, 1 ; open file for writing
                                <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
                                <1>       ; EAX = File Size (= 0)
0000DE4F EB5D                   <1>       jmp    short sysopen_2
                                <1>
                                <1> sysopen: ;<open file>
                                <1>       ; 26/10/2016
                                <1>       ; 24/10/2016
                                <1>       ; 17/10/2016
                                <1>       ; 15/10/2016
                                <1>       ; 06/10/2016, 07/10/2016, 08/10/2016
                                <1>       ; 05/10/2016 (TRDOS 386 = TRDOS v2.0)
                                <1>       ;          -derived from INT 21H.ASM-
                                <1>       ;            ("loc_INT21h_open_file")
                                <1>       ;    26/02/2011
                                <1>       ;    INT 21h Function AH = 3Dh
                                <1>       ;    Open File
                                <1>       ;    INPUT
                                <1>       ;       AL= File Access Value
                                <1>       ;          0- Open for reading
                                <1>       ;            1- Open for writing
                                <1>       ;              2- Open for reading and writing
                                <1>       ;          DS:DX= Pointer to filename (ASCIIZ)
                                <1>       ;
                                <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
                                <1>       ; 22/05/2013 - 27/05/2013 (Retro UNIX 8086 v1)
                                <1>       ;
                                <1>       ; 'sysopen' opens a file in following manner:
                                <1>       ;    1) The second argument in a sysopen says whether to
                                <1>       ;      open the file ro read (0) or write (>0).
                                <1>       ;    2) I-node of the particular file is obtained via 'namei'.
                                <1>       ;    3) The file is opened by 'iopen'.
                                <1>       ;    4) Next housekeeping is performed on the fsp table
                                <1>       ;      and the user's open file list - u.fp.
                                <1>       ;      a) u.fp and fsp are scanned for the next available slot.
                                <1>       ;      b) An entry for the file is created in the fsp table.
                                <1>       ;      c) The number of this entry is put on u.fp list.
                                <1>       ;      d) The file descriptor index to u.fp list is pointed
                                <1>       ;         to by u.r0.
                                <1>       ;
                                <1>       ; Calling sequence:
                                <1>       ;      sysopen; name; mode
                                <1>       ; Arguments:
                                <1>       ;      name - file name or path name
                                <1>       ;      mode - 0 to open for reading
                                <1>       ;             1 to open for writing
                                <1>       ; Inputs: (arguments)
                                <1>       ; Outputs: *u.r0 - index to u.fp list (the file descriptor)
                                <1>       ;              is put into r0's location on the stack.
                                <1>       ; ..........................................................
                                <1>       ;
                                <1>       ; Retro UNIX 8086 v1 modification:
                                <1>       ;       'sysopen' 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>       ;      AX register (will be restored via 'u.r0') will return
                                <1>       ;      to the user with the file descriptor/number
                                <1>       ;      (index to u.fp list).
                                <1>       ;
                                <1>       ;call  arg2
                                <1>       ; * name - 'u.namep' points to address of file/path name
                                <1>       ;          in the user's program segment ('u.segmnt')
                                <1>       ;          with offset in BX register (as sysopen argument 1).
                                <1>       ; * mode - sysopen argument 2 is in CX register
                                <1>       ;          which is on top of stack.
                                <1>       ;
                                <1>       ; jsr r0,arg2 / get sys args into u.namep and on stack
                                <1>       ;
                                <1>            ; system call registers: ebx, ecx (through 'sysenter')
                                <1>       ;
                                <1>       ; TRDOS 386 (05/10/2016)
                                <1>       ;
                                <1>         ; INPUT ->
                                <1>         ;       CL = File Access Value (Open Mode)
                                <1>       ;            0 - Open file for reading
                                <1>       ;              1 - Open file for writing
                                <1>         ;              2 - Open device for reading
                                <1>       ;            3 - Open device for writing
                                <1>         ;          EBX = Pointer to filename/devicename (ASCIIZ)
                                <1>       ; OUTPUT ->
                                <1>       ;          eax = File/Device Handle/Number (index) (AL)
                                <1>       ;          cf = 1 -> Error code in AL
                                <1>       ;
                                <1>       ; Modified Registers: EAX (at the return of system call)
```

```
1618                                    <1>       ;
1619                                    <1>
1620 0000DE51 80F901                    <1>       cmp    cl, 1 ; read file (0), write file (1)
1621 0000DE54 7614                      <1>       jna    short sysopen_0
1622                                    <1>
1623 0000DE56 80F903                    <1>       cmp    cl, 3
1624 0000DE59 0F8640010000              <1>       jna    sysopen_device
1625                                    <1>
1626                                    <1>       ; Invalid access code
1627 0000DE5F B817000000                <1>       mov    eax, ERR_INV_PARAMETER
1628 0000DE64 0F874B010000              <1>       ja     sysopen_dev_err
1629                                    <1>
1630                                    <1> sysopen_0:
1631                                    <1>       ;mov   [u.namep], ebx
1632 0000DE6A 51                        <1>       push   ecx
1633 0000DE6B 89DE                      <1>       mov    esi, ebx
1634                                    <1>       ; file name is forced, change directory as temporary
1635                                    <1>       ;mov   ax, 1
1636                                    <1>       ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
1637                                    <1>       ;call  set_working_path
1638 0000DE6D E851510000                <1>       call   set_working_path_x ; 17/10/2016
1639 0000DE72 731E                      <1>       jnc    short sysopen_1
1640                                    <1>
1641                                    <1> syscreat_err: ; ecx = file attributes (for 'syscreat')
1642 0000DE74 59                        <1>       pop    ecx ; open mode
1643 0000DE75 21C0                      <1>       and    eax, eax  ; 0 -> Bad Path!
1644 0000DE77 7505                      <1>       jnz    short sysopen_err
1645                                    <1>       ; eax = 0
1646 0000DE79 B80C000000                <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
1647                                    <1> sysopen_err:
1648 0000DE7E A3[64030300]              <1>       mov    [u.r0], eax
1649 0000DE83 A3[C8030300]              <1>       mov    [u.error], eax
1650 0000DE88 E80B520000                <1>       call   reset_working_path
1651 0000DE8D E95AFAFFFF                <1>       jmp    error
1652                                    <1>
1653                                    <1> sysopen_1:
1654                                    <1>       ;mov   esi, FindFile_Name
1655 0000DE92 66B80018                  <1>         mov  ax, 1800h ; Only files
1656 0000DE96 E8F6B5FFFF                <1>       call   find_first_file
1657 0000DE9B 5A                        <1>       pop    edx
1658 0000DE9C 72E0                      <1>       jc     short sysopen_err ; eax = 2 (File not found !)
1659                                    <1>
1660                                    <1>       ; check_open_file_attr_access_code
1661                                    <1>
1662 0000DE9E F6C307                    <1>         test bl, 7  ; system, hidden, readonly
1663 0000DEA1 740B                      <1>         jz   short sysopen_2
1664                                    <1>
1665 0000DEA3 20D2                      <1>       and    dl, dl ; 0 = read mode
1666 0000DEA5 7407                      <1>       jz     short sysopen_2
1667                                    <1>
1668                                    <1>       ; 1 = write, 2 = read & write, >2 = invalid
1669 0000DEA7 B80B000000                <1>         mov  eax, ERR_FILE_ACCESS  ; 11 = 'permission denied !'
1670 0000DEAC EBD0                      <1>         jmp  short sysopen_err
1671                                    <1>
1672                                    <1> sysopen_2:
1673                                    <1>       ; esi = Directory Entry (FindFile_DirEntry) Location
1674 0000DEAE 89F3                      <1>       mov    ebx, esi
1675 0000DEB0 31F6                      <1>         xor     esi, esi ; 0
1676 0000DEB2 31FF                      <1>         xor     edi, edi ; 0
1677                                    <1> sysopen_3: ; scan the list of entries in fsp table
1678 0000DEB4 80BE[6A030300]00          <1>         cmp     byte [esi+u.fp], 0
1679 0000DEBB 760F                      <1>         jna     short sysopen_4 ; empty slot
1680 0000DEBD 6646                      <1>         inc     si
1681 0000DEBF 6683FE0A                  <1>         cmp     si, 10
1682 0000DEC3 72EF                      <1>       jb     short sysopen_3
1683                                    <1> toomanyf:
1684 0000DEC5 B80D000000                <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
1685 0000DECA EBB2                      <1>       jmp    short sysopen_err
1686                                    <1>
1687                                    <1> sysopen_4:
1688 0000DECC 80BF[3E9A0100]00          <1>         cmp     byte [edi+OF_MODE], 0 ; Scan open files table
1689 0000DED3 760A                      <1>       jna     short sysopen_5
1690 0000DED5 6647                      <1>       inc    di
1691 0000DED7 6683FF0A                  <1>       cmp     di, OPENFILES ; max. number of open files (=10)
1692 0000DEDB 72EF                      <1>       jb     short sysopen_4
1693 0000DEDD EBE6                      <1>       jmp    short toomanyf
1694                                    <1>
1695                                    <1> sysopen_5:
1696 0000DEDF FEC2                      <1>       inc    dl
1697 0000DEE1 8897[3E9A0100]            <1>         mov     [edi+OF_MODE], dl
1698 0000DEE7 8A15[7E930100]            <1>       mov    dl, [FindFile_Drv]
1699 0000DEED 8897[349A0100]            <1>         mov     [edi+OF_DRIVE], dl ; Logical DOS drive number
1700 0000DEF3 66C1E702                  <1>       shl    di, 2 ; *4 (dword offset)
1701                                    <1>
1702 0000DEF7 8987[849A0100]            <1>       mov    [edi+OF_SIZE], eax ; File size in bytes
1703                                    <1>
1704 0000DEFD 668B4314                  <1>         mov  ax, [ebx+DirEntry_FstClusHI]
1705 0000DF01 C1E010                    <1>       shl    eax, 16
1706 0000DF04 668B431A                  <1>       mov    ax, [ebx+DirEntry_FstClusLO]
1707 0000DF08 8987[0C9A0100]            <1>       mov     [edi+OF_FCLUSTER], eax ; First cluster
1708 0000DF0E 8987[249B0100]            <1>       mov     [edi+OF_CCLUSTER], eax ; Current cluster
1709                                    <1>
1710 0000DF14 31DB                      <1>         xor  ebx, ebx
1711 0000DF16 899F[5C9A0100]            <1>         mov     [edi+OF_POINTER], ebx ; offset pointer (0)
1712 0000DF1C 899F[4C9B0100]            <1>         mov     [edi+OF_CCINDEX], ebx ; cluster index (0)
1713                                    <1>
1714 0000DF22 A1[F0930100]              <1>       mov    eax, [FindFile_DirFirstCluster]
1715 0000DF27 8987[AC9A0100]            <1>       mov    [edi+OF_DIRFCLUSTER], eax
1716                                    <1>
1717 0000DF2D A1[F4930100]              <1>       mov    eax, [FindFile_DirCluster]
1718 0000DF32 8987[D49A0100]            <1>       mov    [edi+OF_DIRCLUSTER], eax
1719                                    <1>
1720                                    <1>       ; Get (& Save) Volume ID
1721                                    <1>       ; Important for files of removable drives
1722                                    <1>       ; (In order to check the drive has same volume/disk)
```

```
1723 0000DF38 88D7              <1>       mov    bh, dl
1724 0000DF3A 81C300010900      <1>         add  ebx, Logical_DOSDisks
1725 0000DF40 8A4303            <1>         mov  al, [ebx+LD_FATType]
1726 0000DF43 3C01              <1>         cmp  al, 1
1727 0000DF45 7209              <1>         jb   short sysopen_6_fs
1728 0000DF47 3C02              <1>         cmp  al, 2
1729 0000DF49 770A              <1>         ja   short sysopen_6_fat32
1730                            <1> sysopen_6_fat:
1731 0000DF4B 8B432D            <1>         mov  eax, [ebx+LD_BPB+VolumeID]
1732 0000DF4E EB08              <1>         jmp  short sysopen_7
1733                            <1> sysopen_6_fs:
1734 0000DF50 8B4328            <1>         mov  eax, [ebx+LD_FS_VolumeSerial]
1735 0000DF53 EB03              <1>         jmp  short sysopen_7
1736                            <1> sysopen_6_fat32:
1737 0000DF55 8B4349            <1>         mov  eax, [ebx+LD_BPB+FAT32_VolID]
1738                            <1> sysopen_7:
1739 0000DF58 A3[D48A0100]      <1>         mov  [Current_VolSerial], eax
1740                            <1>
1741 0000DF5D 8987[FC9A0100]    <1>       mov    [edi+OF_VOLUMEID], eax
1742                            <1>
1743                            <1>       ; 24/10/2016
1744 0000DF63 66D1EF            <1>       shr    di, 1 ; 4/2, word offset
1745 0000DF66 668B1D[F8930100]  <1>       mov    bx, [FindFile_DirEntryNumber]
1746 0000DF6D 66899F[749B0100]  <1>       mov    [edi+OF_DIRENTRY], bx
1747                            <1>
1748 0000DF74 31D2              <1>       xor    edx, edx
1749                            <1>       ;shr   di, 2 ; /4 (byte offset)
1750 0000DF76 66D1EF            <1>       shr    di, 1 ; 2/2, byte offset
1751 0000DF79 8897[529A0100]    <1>       mov    byte [edi+OF_OPENCOUNT], dl ; 0
1752 0000DF7F 8897[489A0100]    <1>       mov    byte [edi+OF_STATUS], dl ; 0
1753                            <1>
1754 0000DF85 89FB              <1>       mov    ebx, edi
1755 0000DF87 FEC3              <1>       inc    bl
1756                            <1>
1757 0000DF89 889E[6A030300]    <1>         mov     [esi+u.fp], bl ; Open File Entry Number
1758 0000DF8F 8935[64030300]    <1>       mov     [u.r0], esi ; move index to u.fp list
1759                            <1>                         ; into eax on stack
1760                            <1>
1761 0000DF95 E8FE500000        <1>         call       reset_working_path
1762                            <1>
1763 0000DF9A E96DF9FFFF        <1>       jmp    sysret
1764                            <1>
1765                            <1>       ; (Retro UNIX 386 v1.0)
1766                            <1>       ; 'fsp' table (10 bytes/entry)
1767                            <1>       ; bit 15                                bit 0
1768                            <1>       ; ---|-----------------------------------------
1769                            <1>       ; r/w|       i-number of open file
1770                            <1>       ; ---|-----------------------------------------
1771                            <1>       ;               device number
1772                            <1>       ; -----------------------------------------------
1773                            <1>       ; offset pointer, r/w pointer to file (bit 0-15)
1774                            <1>       ; -----------------------------------------------
1775                            <1>       ; offset pointer, r/w pointer to file (bit 16-31)
1776                            <1>       ; ----------------------|------------------------
1777                            <1>       ;  flag that says file    | number of processes
1778                            <1>       ;   has been deleted      | that have file open
1779                            <1>       ; ----------------------|------------------------
1780                            <1>
1781                            <1> sysopen_device:
1782                            <1>       ; 15/10/2016
1783                            <1>       ; 08/10/2016
1784                            <1>       ; 07/10/2016 (TRDOS 386 = TRDOS v2.0)
1785 0000DF9F 51                <1>       push   ecx ; open mode
1786 0000DFA0 89E5              <1>       mov    ebp, esp
1787 0000DFA2 B910000000        <1>       mov    ecx, 16 ; transfer length = 16 bytes
1788 0000DFA7 29CC              <1>       sub    esp, ecx
1789 0000DFA9 89E7              <1>       mov    edi, esp ; destination address
1790 0000DFAB 89DE              <1>       mov    esi, ebx ; dev name in user's memory space
1791 0000DFAD E8DF3B0000        <1>       call   transfer_from_user_buffer
1792 0000DFB2 7310              <1>       jnc    short sysopen_dev_0
1793                            <1>       ; eax = ERR_OUT_OF_MEMORY = 4 = ERR_MINOR_IM
1794 0000DFB4 59                <1>       pop    ecx
1795                            <1> sysopen_dev_err:
1796 0000DFB5 A3[64030300]      <1>       mov    [u.r0], eax
1797 0000DFBA A3[C8030300]      <1>       mov    [u.error], eax
1798 0000DFBF E928F9FFFF        <1>       jmp    error
1799                            <1> sysopen_dev_0:
1800 0000DFC4 89FE              <1>       mov    esi, edi ; Device name addr (max. 16 bytes, ASCIIZ)
1801                            <1>                       ; for example: "tty, TTY, /dev/tty"
1802 0000DFC6 E873530000        <1>       call   get_device_number
1803 0000DFCB 89EC              <1>       mov    esp, ebp
1804 0000DFCD 59                <1>       pop    ecx
1805 0000DFCE 7307              <1>       jnc    short sysopen_dev_1
1806 0000DFD0 B818000000        <1>       mov    eax, ERR_INV_DEV_NAME ; 24 ; 'invalid device name !'
1807 0000DFD5 EBDE              <1>       jmp    short sysopen_dev_err
1808                            <1> sysopen_dev_1:
1809                            <1>       ; eax = Device Number (AL)
1810                            <1>       ;  cl = Open mode (2 = device read, 3 = device write)
1811 0000DFD7 31DB              <1>         xor     ebx, ebx ; 0
1812                            <1> sysopen_dev_2: ; scan the list of entries
1813 0000DFD9 389B[6A030300]    <1>         cmp     [ebx+u.fp], bl ; 0
1814 0000DFDF 760E              <1>         jna     short sysopen_dev_3 ; empty slot
1815 0000DFE1 FEC3              <1>         inc     bl
1816 0000DFE3 80FB0A            <1>         cmp     bl, 10
1817 0000DFE6 72F1              <1>       jb     short sysopen_dev_2
1818                            <1>       ;
1819 0000DFE8 B80D000000        <1>       mov    eax, ERR_TOO_MANY_FILES ; too many open files !
1820 0000DFED EBC6              <1>       jmp    short sysopen_dev_err
1821                            <1> sysopen_dev_3:
1822 0000DFEF 891D[64030300]    <1>       mov    [u.r0], ebx ; File/Device index/handle/descriptor
1823                            <1>       ; eax = device number (entry offset)
1824 0000DFF5 8AA8[D0970100]    <1>       mov    ch, [eax+DEV_ACCESS] ; bit 0 = accessable by users
1825                            <1>                                   ; bit 1 = read access perm
1826                            <1>                                   ; bit 2 = write access perm
1827                            <1>                                   ; bit 3 = IOCTL permit to users
```

```
1828                            <1>                                 ; bit 4 = block device if set
1829                            <1>                                 ; bit 5 = 16 bit or 1024 byte
1830                            <1>                                 ; bit 6 = 32 bit or 2048 byte
1831                            <1>                                 ; bit 7 = installable device drv
1832 0000DFFB F6C501           <1>       test   ch, 1 ; accessable by normal users (except root)
1833 0000DFFE 7510             <1>       jnz    short sysopen_dev_4 ; yes, permission has been given
1834 0000E000 803D[B0030300]00 <1>       cmp    byte [u.uid], 0 ; root?
1835 0000E007 7607             <1>       jna    short sysopen_dev_4 ; superuser can open all devices
1836                            <1> sysopen_dev_perm_err:
1837 0000E009 B80B000000       <1>       mov    eax, ERR_DEV_ACCESS  ; 11 = 'permission denied !'
1838 0000E00E EBA5             <1>       jmp    short sysopen_dev_err
1839                            <1> sysopen_dev_4:
1840 0000E010 D0ED             <1>       shr    ch, 1 ; result: 1 = read, 2 = write, 3 = r & w
1841 0000E012 FEC9             <1>       dec    cl  ; result: 1 = read, 2 = write
1842 0000E014 84E9             <1>       test   cl, ch
1843 0000E016 74F1             <1>       jz     short sysopen_dev_perm_err
1844                            <1>
1845 0000E018 D0E5             <1>       shl    ch, 1 ; bit 0 = 0
1846                            <1>       ; eax = device number (entry offset)
1847 0000E01A E83B540000       <1>       call   device_open
1848 0000E01F 72E8             <1>       jc     short sysopen_dev_perm_err
1849                            <1>
1850                            <1>       ; eax = device number (entry offset)
1851 0000E021 0C80             <1>       or     al, 80h ; set device bit (set bit 7 to 1)
1852 0000E023 8B1D[64030300]   <1>       mov    ebx, [u.r0]
1853 0000E029 8883[6A030300]   <1>       mov    [ebx+u.fp], al      ; bit 7 (=1) points to device
1854                            <1>
1855 0000E02F E9D8F8FFFF       <1>       jmp    sysret
1856                            <1>
1857                            <1> sysmkdir: ; < make directory >
1858                            <1>       ; 15/10/2016
1859                            <1>       ; 10/10/2016 (TRDOS 386 = TRDOS v2.0)
1860                            <1>       ;          -derived from INT 21H.ASM-
1861                            <1>       ;            ("loc_INT21h_create_file")
1862                            <1>         ;    10/07/2011 (12/03/2011)
1863                            <1>         ;    INT 21h Function AH = 3Ch
1864                            <1>         ;    Create File
1865                            <1>         ;    INPUT
1866                            <1>         ;       CX = Attributes
1867                            <1>         ;           DS:DX= Address of zero terminaned path name
1868                            <1>         ;
1869                            <1>         ;
1870                            <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
1871                            <1>       ; 27/05/2013 - 02/08/2013 (Retro UNIX 8086 v1)
1872                            <1>       ;
1873                            <1>       ; 'sysmkdir' creates an empty directory whose name is
1874                            <1>       ; pointed to by arg 1. The mode of the directory is arg 2.
1875                            <1>       ; The special entries '.' and '..' are not present.
1876                            <1>       ; Errors are indicated if the directory already exists or
1877                            <1>       ; user is not the super user.
1878                            <1>       ;
1879                            <1>       ; Calling sequence:
1880                            <1>       ;     sysmkdir; name; mode
1881                            <1>       ; Arguments:
1882                            <1>       ;     name - points to the name of the directory
1883                            <1>       ;     mode - mode of the directory
1884                            <1>       ; Inputs: (arguments)
1885                            <1>       ; Outputs: -
1886                            <1>       ;    (sets 'directory' flag to 1;
1887                            <1>       ;    'set user id on execution' and 'executable' flags to 0)
1888                            <1>       ; ..........................................................
1889                            <1>       ;
1890                            <1>       ; Retro UNIX 8086 v1 modification:
1891                            <1>       ;       'sysmkdir' system call has two arguments; so,
1892                            <1>       ;     * 1st argument, name is pointed to by BX register
1893                            <1>       ;     * 2nd argument, mode is in CX register
1894                            <1>       ;
1895                            <1>       ; TRDOS 386 (10/10/2016)
1896                            <1>       ;
1897                            <1>         ; INPUT ->
1898                            <1>         ;       CL = Directory Attributes
1899                            <1>       ;           bit 0 (1) - Read only file/dir (R)
1900                            <1>       ;             bit 1 (1) - Hidden file/dir (H)
1901                            <1>         ;             bit 2 (1) - System file/dir (R)
1902                            <1>       ;             bit 3 (1) - Volume label/name (V)
1903                            <1>         ;             bit 4 (1) - Subdirectory (D)
1904                            <1>       ;           bit 5 (1) - File/Dir has been archived (A)
1905                            <1>       ;         CX = 0 -> create normal directory
1906                            <1>         ;          EBX = Pointer to directory name (ASCIIZ) -path-
1907                            <1>       ;
1908                            <1>       ; OUTPUT ->
1909                            <1>       ;          eax = First cluster of the new directory
1910                            <1>       ;          cf = 1 -> Error code in AL
1911                            <1>       ;
1912                            <1>       ; Modified Registers: EAX (at the return of system call)
1913                            <1>       ;
1914                            <1>       ; Note: If the file or directory is existing
1915                            <1>       ;     an access error will be returned.
1916                            <1>
1917 0000E034 6621C9           <1>       and    cx, cx ; if cx = 0 -> create a normal subdir
1918 0000E037 7413             <1>       jz     short sysmkdir_1
1919                            <1>
1920 0000E039 F6C110           <1>       test   cl, 10h ; if dir flags set, also use other flags
1921 0000E03C 0F853EFDFFFF     <1>       jnz    sysmkdir_0 ; jump to head of 'syscreat'
1922                            <1>
1923                            <1>       ; CX has wrong flags
1924 0000E042 B817000000       <1>       mov    eax, ERR_INV_FLAGS
1925 0000E047 E969FFFFFF       <1>       jmp    sysopen_dev_err
1926                            <1>
1927                            <1> sysmkdir_1:
1928 0000E04C B110             <1>       mov    cl, 10h ; set subdir flag and reset other flags
1929 0000E04E E92DFDFFFF       <1>       jmp    sysmkdir_0 ; jump to head of 'syscreat'
1930                            <1> sysmkdir_2:
1931                            <1>       ; jump from 'syscreat' ; from 'syscreat_1'
1932                            <1>       ;  CL = Directory attributes/flags
```

```
1933 0000E053 BE[C0930100]        <1>       mov    esi, FindFile_Name
1934 0000E058 E804D7FFFF          <1>       call   make_sub_directory
1935 0000E05D 0F821BFEFFFF        <1>       jc     sysopen_err       ; NOTE: Old type (TRDOS 8086)
1936                              <1>                                ; error codes must be modified
1937                              <1>                                ; for next TRDOS 386 versions
1938                              <1>                                ; (10/10/2016)
1939                              <1>                                ; Old (MSDOS type)
1940                              <1>                                ; error codes (2011):
1941                              <1>                                ;  2 = file not found
1942                              <1>                                ;  3 = directory not found
1943                              <1>                                ;  5 = access denied
1944                              <1>                                ; 12 = no more files
1945                              <1>                                  ; 19 = disk write protected
1946                              <1>                                ; 39 = insufficient disk space
1947                              <1>                                ; 'sysdefs.s' ; 10/10/2016
1948                              <1>
1949 0000E063 A3[64030300]        <1>       mov    [u.r0], eax ; New sub dir's first cluster
1950                              <1>
1951 0000E068 E82B500000          <1>         call       reset_working_path
1952                              <1>
1953 0000E06D E99AF8FFFF          <1>       jmp    sysret
1954                              <1>
1955                              <1> sysclose: ;<close file>
1956                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
1957                              <1>       ;
1958                              <1>       ; 14/05/2015 (Retro UNIX 386 v1 - Beginning)
1959                              <1>       ; 22/05/2013 - 26/05/2013 (Retro UNIX 8086 v1)
1960                              <1>       ;
1961                              <1>       ; 'sysclose', given a file descriptor in 'u.r0', closes the
1962                              <1>       ; associated file. The file descriptor (index to 'u.fp' list)
1963                              <1>       ; is put in r1 and 'fclose' is called.
1964                              <1>       ;
1965                              <1>       ; Calling sequence:
1966                              <1>       ;     sysclose
1967                              <1>       ; Arguments:
1968                              <1>       ;     -
1969                              <1>       ; Inputs: *u.r0 - file descriptor
1970                              <1>       ; Outputs: -
1971                              <1>       ; ..........................................................
1972                              <1>       ;
1973                              <1>       ; Retro UNIX 8086 v1 modification:
1974                              <1>       ;      The user/application program puts file descriptor
1975                              <1>       ;       in BX register as 'sysclose' system call argument.
1976                              <1>       ;      (argument transfer method 1)
1977                              <1>
1978                              <1>       ; TRDOS 386 (06/10/2016)
1979                              <1>       ;
1980                              <1>         ; INPUT ->
1981                              <1>         ;      EBX = File Handle/Number (file index) (AL)
1982                              <1>       ; OUTPUT ->
1983                              <1>       ;          cf = 0 -> EAX = 0
1984                              <1>       ;          cf = 1 -> Error code in EAX (ERR_FILE_NOT_OPEN)
1985                              <1>       ;
1986                              <1>       ; Modified Registers: EAX (at the return of system call)
1987                              <1>       ;
1988                              <1>
1989 0000E072 89D8                <1>       mov    eax, ebx
1990 0000E074 31DB                <1>       xor    ebx, ebx
1991 0000E076 891D[64030300]      <1>       mov    [u.r0], ebx ; 0  ; return value of EAX
1992 0000E07C E89F2F0000          <1>       call   fclose
1993 0000E081 0F8385F8FFFF        <1>       jnc    sysret
1994 0000E087 B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
1995 0000E08C A3[C8030300]        <1>       mov    [u.error], eax ;
1996 0000E091 A3[64030300]        <1>       mov    [u.r0], eax ; ! invalid handle !
1997 0000E096 E951F8FFFF          <1>       jmp    error
1998                              <1>
1999                              <1> sysread: ; < read from file >
2000                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2001                              <1>       ;          -derived from INT_21H.ASM-
2002                              <1>       ;            ("loc_INT21h_read_file")
2003                              <1>         ;    13/03/2011 (05/03/2011)
2004                              <1>         ;    INT 21h Function AH = 3Fh
2005                              <1>         ;    Read from a File
2006                              <1>         ;    INPUT
2007                              <1>       ;         BX = File Handle
2008                              <1>         ;       CX = Number of bytes to read
2009                              <1>         ;           DS:DX= Buffer address
2010                              <1>         ;
2011                              <1>       ; Note: TRDOS 386 'sysread' has been derived from
2012                              <1>       ;     Retro UNIX 386 v1 'sysread', except a few
2013                              <1>       ;     code modifications.
2014                              <1>       ;
2015                              <1>       ; 13/05/2015 (Retro UNIX 386 v1)
2016                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
2017                              <1>       ; 23/05/2013 (Retro UNIX 8086 v1)
2018                              <1>       ;
2019                              <1>       ; 'sysread' is given a buffer to read into and the number of
2020                              <1>       ; characters to be read. If finds the file from the file
2021                              <1>       ; descriptor located in *u.r0 (r0). This file descriptor
2022                              <1>       ; is returned from a successful open call (sysopen).
2023                              <1>       ; The i-number of file is obtained via 'rw1' and the data
2024                              <1>       ; is read into core via 'readi'.
2025                              <1>       ;
2026                              <1>       ; Calling sequence:
2027                              <1>       ;     sysread; buffer; nchars
2028                              <1>       ; Arguments:
2029                              <1>       ;     buffer - location of contiguous bytes where
2030                              <1>       ;             input will be placed.
2031                              <1>       ;     nchars - number of bytes or characters to be read.
2032                              <1>       ; Inputs: *u.r0 - file descriptor (& arguments)
2033                              <1>       ; Outputs: *u.r0 - number of bytes read.
2034                              <1>       ; ..........................................................
2035                              <1>       ;
2036                              <1>       ; Retro UNIX 8086 v1 modification:
2037                              <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
0000E09B E8CE2F0000     <1>       call   getf1
0000E0A0 7277           <1>       jc     short device_read ; read data from device
                        <1>       ; EAX = First cluster of the file
                        <1>
0000E0A2 E83F000000     <1>       call   rw1
0000E0A7 730A           <1>       jnc    short sysread_0
                        <1>
0000E0A9 A3[64030300]   <1>       mov    [u.r0], eax ; error code
0000E0AE E939F8FFFF     <1>       jmp    error
                        <1>
                        <1> sysread_0:
0000E0B3 E8C5350000     <1>       call   readi
0000E0B8 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
0000E0BA E8AF2F0000     <1>       call   getf1
0000E0BF 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>
0000E0C1 E820000000     <1>       call   rw1
0000E0C6 730A           <1>       jnc    short syswrite_0
0000E0C8 A3[64030300]   <1>       mov    [u.r0], eax ; error code
0000E0CD E91AF8FFFF     <1>       jmp    error
                        <1>
                        <1> syswrite_0:
```

```
2143 0000E0D2 E8D23C0000          <1>       call   writei
2144                              <1> rw0: ; 1:
2145 0000E0D7 A1[8C030300]        <1>         mov  eax, [u.nread]
2146 0000E0DC A3[64030300]        <1>       mov    [u.r0], eax
2147 0000E0E1 E926F8FFFF          <1>       jmp    sysret
2148                              <1>
2149                              <1> rw1:
2150                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2151                              <1>       ; 14/05/2015 (Retro UNIX 386 v1)
2152                              <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
2153                              <1>       ; 23/05/2013 - 24/05/2013 (Retro UNIX 8086 v1)
2154                              <1>       ; System call registers: ebx, ecx, edx (through 'sysenter')
2155                              <1>       ;
2156                              <1>       ; EBX = File descriptor
2157                              <1>       ;call  getf1 ; calling point in 'getf' from 'rw1'
2158                              <1>       ;jc    short device_rw ; read/write data from/to device
2159                              <1>       ; EAX = First cluster of the file
2160                              <1>
2161 0000E0E6 83F802              <1>       cmp    eax, 2
2162 0000E0E9 7217                <1>       jb     short rw2
2163                              <1>       ;
2164 0000E0EB 890D[84030300]      <1>       mov    [u.base], ecx       ; buffer address/offset
2165                              <1>                                  ;(in the user's virtual memory space)
2166 0000E0F1 8915[88030300]      <1>       mov    [u.count], edx
2167                              <1>
2168 0000E0F7 C705[C8030300]0000- <1>         mov     dword [u.error], 0 ; reset the last error code
2168 0000E0FF 0000                <1>
2169 0000E101 C3                  <1>       retn
2170                              <1>
2171                              <1> rw2:
2172 0000E102 B80A000000          <1>       mov    eax, ERR_FILE_NOT_OPEN ; file not open !
2173 0000E107 A3[C8030300]        <1>       mov    dword [u.error], eax
2174 0000E10C C3                  <1>       retn
2175                              <1> rw3:
2176 0000E10D B80B000000          <1>       mov    eax, ERR_FILE_ACCESS ; permission denied !
2177 0000E112 A3[C8030300]        <1>       mov    dword [u.error], eax
2178 0000E117 F9                  <1>       stc
2179 0000E118 C3                  <1>       retn
2180                              <1>
2181                              <1> device_read:
2182                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2183                              <1>       ; cl = DEV_OPENMODE ; open mode
2184                              <1>       ; ch = DEV_ACCESS   ; access flags
2185                              <1>       ; al = DEV_DRIVER   ; device number (eax)
2186                              <1>
2187 0000E119 F6C101              <1>       test   cl, 1 ; 1 = read, 2 = write, 3 = read&write
2188 0000E11C 74EF                <1>       jz     short rw3
2189                              <1>
2190 0000E11E 89C3                <1>       mov    ebx, eax
2191 0000E120 66C1E302            <1>       shl    bx, 2 ; *4
2192                              <1>
2193 0000E124 F6C580              <1>       test   ch, 80h ; bit 7, installable device driver flag
2194 0000E127 7406                <1>       jz     short d_read_2 ; Kernel device
2195                              <1>       ; installable device
2196                              <1> d_read_1:
2197 0000E129 FFA3[8C970100]      <1>         jmp  dword [ebx+IDEV_RADDR-4]
2198                              <1> d_read_2:
2199 0000E12F FFA3[22490100]      <1>       jmp    dword [ebx+KDEV_RADDR-4]
2200                              <1>
2201                              <1> device_write:
2202                              <1>       ; 11/10/2016 (TRDOS 386 = TRDOS v2.0)
2203                              <1>       ; cl = DEV_OPENMODE ; open mode
2204                              <1>       ; ch = DEV_ACCESS   ; access flags
2205                              <1>       ; al = DEV_DRIVER   ; device number (eax)
2206                              <1>
2207 0000E135 F6C102              <1>       test   cl, 2 ; 1 = read, 2 = write, 3 = read&write
2208 0000E138 74D3                <1>       jz     short rw3
2209                              <1>
2210 0000E13A 89C3                <1>       mov    ebx, eax
2211 0000E13C 66C1E302            <1>       shl    bx, 2 ; *4
2212                              <1>
2213 0000E140 F6C580              <1>       test   ch, 80h ; bit 7, installable device driver flag
2214 0000E143 7406                <1>       jz     short d_write_2 ; Kernel device
2215                              <1>       ; installable device
2216                              <1> d_write_1:
2217 0000E145 FFA3[AC970100]      <1>         jmp  dword [ebx+IDEV_WADDR-4]
2218                              <1> d_write_2:
2219 0000E14B FFA3[72490100]      <1>       jmp    dword [ebx+KDEV_WADDR-4]
2220                              <1>
2221                              <1>
2222                              <1> sysemt: ; enable (or disable) multi tasking -time sharing-
2223                              <1>       ;
2224                              <1>       ; 23/05/2016 - TRDOS 386 (TRDOS v2.0)
2225                              <1>       ; 14/05/2015 (Retro UNIX 386 v1)
2226                              <1>       ; 10/12/2013 - 20/04/2014 (Retro UNIX 8086 v1)
2227                              <1>       ;
2228                              <1>       ; Retro UNIX 8086 v1 modification:
2229                              <1>       ;     'Enable Multi Tasking'  system call instead
2230                              <1>       ;     of 'Emulator Trap' in original UNIX v1 for PDP-11.
2231                              <1>       ;
2232                              <1>       ; Retro UNIX 8086 v1 feature only!
2233                              <1>       ;     Using purpose: Kernel will start without time-out
2234                              <1>       ;     (internal clock/timer) functionality.
2235                              <1>       ;     Then etc/init will enable clock/timer for
2236                              <1>       ;     multi tasking.
2237                              <1>       ;
2238                              <1>       ; INPUT ->
2239                              <1>       ;     BL = 0 -> disable multi tasking
2240                              <1>       ;     BL > 1 -> enable multi tasking (time sharing)
2241                              <1>       ; OUTPUT ->
2242                              <1>       ;     none
2243                              <1>       ;
2244                              <1>       ;  Note: Multi tasking is disabled during system
2245                              <1>       ;       initialization, it must be enabled by using
2246                              <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>
0000E151 803D[B0030300]00       <1>       cmp    byte [u.uid], 0 ; root ?
                                <1>       ;ja    error
0000E158 0F87D3F8FFFF           <1>       ja     badsys ; 14/05/2015
                                <1>       ;
0000E15E FA                     <1>       cli
0000E15F 881D[AA960100]         <1>       mov    [multi_tasking], bl ; 0 to disable, >0 to enable
0000E165 E9A2F7FFFF             <1>       jmp    sysret
                                <1>
                                <1> systimer:
                                <1>       ; 02/01/2017
                                <1>       ; 21/12/2016
                                <1>       ; 19/12/2016
                                <1>       ; 10/12/2016 (callback)
                                <1>       ; 10/06/2016
                                <1>       ; 07/06/2016
                                <1>       ; 06/06/2016
                                <1>       ; 21/05/2016
                                <1>       ; 19/05/2016
                                <1>       ; 18/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; (TRDOS 386 feature only!)
                                <1>       ;
                                <1>       ; (start or stop timer event(s))
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     BL = Signal return byte (response byte)
                                <1>       ;          (Any requested value between 0 and 255)
                                <1>       ;          (Kernel will put it at the requested address)
                                <1>       ;     BH = Time count unit
                                <1>       ;          0 = Stop timer event
                                <1>       ;          1 = 18.2 ticks per second
                                <1>       ;          2 = 10 milliseconds
                                <1>       ;          3 = 1 second (for real time clock interrupt)
                                <1>       ;          4 = time/tick count in current time count unit
                                <1>       ;         // 10/12/2016
                                <1>       ;         80h = Stop timer event (callback method)
                                <1>       ;         81h = 18.2 ticks per second, callback method
                                <1>       ;         82h = 10 milliseconds, callback method
                                <1>       ;         83h = 1 second (for RTC int), callback method
                                <1>       ;         84h = current time count unit, callback method
                                <1>       ;
                                <1>       ;         Note: Only 03h or 83h will set real time clock
                                <1>       ;              (RTC) events (Others are for PIT events)!
                                <1>       ;
                                <1>       ;     NOTE: If callback (user service) method is used,
                                <1>       ;         EDX will point to the return address (of service
                                <1>       ;         procedure) in user's space instead of signal
                                <1>       ;         response byte address. (TRDOS 386 kernel will
                                <1>       ;         direct the cpu to that address -in user's space-
                                <1>       ;         at the return of system call or interrupt
                                <1>       ;         just after the adjusted count/time is elapsed.)
                                <1>       ;         User's sevice routine must be ended with a
                                <1>       ;         'iret'. Normal return addresses from system
                                <1>       ;         calls or and interrupts will be kept same except
                                <1>       ;         the timer returns.
                                <1>       ;
                                <1>       ;     BH = 0 -> Stop timer event
                                <1>       ;     BL = Timer event number (1 to 255) if BH = 0
                                <1>       ;          If BL = 0, all timer events (which are belongs
                                <1>       ;           to running process) will be stopped
                                <1>       ;     ECX = Time/Tick count (depending on time count unit)
                                <1>       ;     EDX = Signal return (Response) byte address
                                <1>       ;           (virtual address in user's memory space)
                                <1>       ; OUTPUT ->
                                <1>       ;     AL = Timer event number   (1 to 255) (max. value = 16)
                                <1>       ;     IF BH Input = 0 & CF = 0 & AL = 0 ->
                                <1>       ;          timer event(s) has/have been stopped/finished
                                <1>       ;     CF = 1 & AL = 0 -> no timer setting space to set
                                <1>       ;     CF = 1 & AL > 0 -> timer count unit is not usable
                                <1>       ;
                                <1>       ;     NOTE: To modify a time count for a user function,
                                <1>       ;           at first, current timer event must be stopped
                                <1>       ;           then a new timer event (which is related with
                                <1>       ;           same user function) must be started.
                                <1>       ;
                                <1>       ;           Signal return (response) byte may be used for
                                <1>       ;           several purposes. Kernel will put this value
                                <1>       ;           to requested address during timer interrupt,
                                <1>       ;           program/user can check this value to understand
                                <1>       ;           which event has been occurred and what is changed.
                                <1>       ;           (Multi timer events can share same signal address)
                                <1>       ;
                                <1>       ;     NOTE: If the process is running while the time count
                                <1>       ;           is reached, kernel will put signal return (response)
                                <1>       ;           byte value at requested address during timer
                                <1>       ;           interrupt and the process will continue to run.
                                <1>       ;           Program/process must call (jump to) it's timer event
                                <1>       ;           function as required, for checking the timer event
                                <1>       ;           status via signal return (response) byte address.
                                <1>       ;
                                <1>       ;           If the process is not running (waiting or sleeping
                                <1>       ;           or released) while the time count is reached,
                                <1>       ;           it is restarted from where it left, to ensure
                                <1>       ;           proper multi media (video, audio, clock, timer)
                                <1>       ;           functionality.
                                <1>       ;
                                <1>       ;           (It is better to use 'syswait' or 'syssleep',
                                <1>       ;           or 'sysrele' system call just after the timer
                                <1>       ;           function. Otherwise, timer events may block other
```

```
2352                                    <1>       ;           processes which are not using timer events.)

2353                                    <1>       ;
2354                                    <1>       ; Timer Event Structure: (max. 16 timer events, 16*16 bytes)
2355                                    <1>       ;       Owner:              resb 1 ; 0 = free
2356                                    <1>       ;                          ;>0 = process number (u.uno)
2357                                    <1>       ;      Calback:     resb 1 ; 1 = callback, 0 = response byte
2358                                    <1>       ;      Interrupt:      resb 1 ; 0 = Timer interrupt (or none)
2359                                    <1>       ;                          ; 1 = Real Time Clock interrupt
2360                                    <1>       ;      Response:       resb 1 ; 0 to 255, signal return value
2361                                    <1>       ;      Count Limit: resd 1 ; count of ticks (total/set)
2362                                    <1>       ;      Current Count:      resd 1 ; count of ticks (current)
2363                                    <1>       ;      Response Addr:  resd 1 ; response byte (pointer) address
2364                                    <1>       ;
2365                                    <1>
2366                                    <1>       ; 19/12/2016 (timer callback)
2367 0000E16A C605[E89B0100]00         <1>       mov    byte [tcallback], 0
2368 0000E171 C605[E99B0100]00         <1>       mov    byte [trtc], 0
2369 0000E178 C705[D0030300]0000-      <1>       mov    dword [u.tcb], 0 ; this is not necessary...
2369 0000E180 0000                     <1>
2370                                    <1>
2371 0000E182 80FF80                   <1>       cmp    bh, 80h
2372 0000E185 7225                     <1>       jb     short systimer_cb2
2373 0000E187 7704                     <1>       ja     short systimer_cb0
2374                                    <1>
2375 0000E189 31D2                     <1>       xor    edx, edx ; 0, reset callback address
2376 0000E18B EB0B                     <1>       jmp    short systimer_cb1
2377                                    <1>
2378                                    <1> systimer_cb0:
2379 0000E18D 80FF84                   <1>       cmp    bh, 84h
2380 0000E190 7764                     <1>       ja     short systimer_5 ;  undefined, error
2381                                    <1>
2382                                    <1>       ;mov   byte [tcallback], 1 ; 19/12/2016
2383 0000E192 FE05[E89B0100]           <1>       inc    byte [tcallback]
2384                                    <1>
2385                                    <1> systimer_cb1:
2386 0000E198 0FB635[B3030300]         <1>       movzx  esi, byte [u.uno] ; process number
2387 0000E19F 66C1E602                 <1>       shl    si, 2
2388 0000E1A3 8996[0C010300]           <1>       mov    [esi+p.tcb-4], edx ; set process timer callback address
2389                                    <1>                               ; (overwrite prev value if it is set!)
2390 0000E1A9 80E77F                   <1>       and    bh, 7Fh
2391                                    <1>
2392                                    <1> systimer_cb2:
2393 0000E1AC 80FF02                   <1>       cmp    bh, 2
2394 0000E1AF 7445                     <1>         je     short systimer_5  ; only 18.2 ticks per second is usable
2395                                    <1>                               ; 10 milliseconds (100 Hertz) timer
2396                                    <1>                               ; will be set later (18/05/2016)
2397 0000E1B1 774B                     <1>         ja     short systimer_6
2398                                    <1>
2399 0000E1B3 20FF                     <1>       and    bh, bh
2400 0000E1B5 0F84BA000000             <1>         jz     systimer_9        ; stop timer event(s)
2401                                    <1>
2402                                    <1>       ; bh = 1 (timer interrupt, 18.2 Hz, IBM PC/AT ROMBIOS default)
2403                                    <1>
2404                                    <1> systimer_19:
2405 0000E1BB B00A                     <1>       mov    al, 10 ; (*)
2406                                    <1>
2407                                    <1> systimer_0:
2408 0000E1BD B710                     <1>       mov    bh, 16
2409                                    <1>       ;
2410 0000E1BF 383D[AB960100]           <1>       cmp    [timer_events], bh ; 16 ; 07/06/2016
2411 0000E1C5 7319                     <1>       jnb    short systimer_3  ; max. 16 timer events
2412                                    <1>       ;
2413 0000E1C7 50                       <1>       push   eax ; (*)
2414                                    <1>
2415 0000E1C8 BF[60040300]             <1>       mov    edi, timer_set  ; beginning address of timer events
2416                                    <1>                              ; setting space
2417 0000E1CD 30C0                     <1>       xor    al, al ; 0
2418                                    <1> systimer_1:
2419 0000E1CF FEC0                     <1>       inc    al
2420 0000E1D1 803F00                   <1>       cmp    byte [edi], 0       ; is it free space ?
2421 0000E1D4 7639                     <1>       jna    short systimer_7 ; yes
2422 0000E1D6 FECF                     <1>       dec    bh
2423 0000E1D8 7405                     <1>       jz     short systimer_2
2424 0000E1DA 83C710                   <1>       add    edi, 16
2425 0000E1DD EBF0                     <1>       jmp    short  systimer_1 ; next event space
2426                                    <1>
2427                                    <1> systimer_2:
2428 0000E1DF 58                       <1>       pop    eax ; (*) discard
2429                                    <1> systimer_3:
2430 0000E1E0 C605[64030300]00         <1>       mov    byte [u.r0], 0
2431                                    <1> systimer_4:
2432 0000E1E7 C705[C8030300]1B00-      <1>         mov     dword [u.error], ERR_MISC
2432 0000E1EF 0000                     <1>
2433                                    <1>                                     ; one of miscellaneous/other errors
2434 0000E1F1 E9F6F6FFFF               <1>       jmp    error ; cf -> 1
2435                                    <1>
2436                                    <1> systimer_5:
2437 0000E1F6 883D[64030300]           <1>       mov    [u.r0], bh ; Time count unit (=2 or >3)
2438 0000E1FC EBE9                     <1>       jmp    short systimer_4 ; 07/06/2016
2439                                    <1>
2440                                    <1> systimer_6:
2441 0000E1FE 80FF04                   <1>       cmp    bh, 4
2442 0000E201 77F3                     <1>         ja     short systimer_5  ; undefined time count unit
2443                                    <1>       ;jb    short systimer_16
2444                                    <1>
2445                                    <1>       ;mov   al, 1  ; default (use current timer unit)
2446                                    <1>                     ; countdown value is in ECX !
2447                                    <1>                     ; max. value of ecx = 4294967296/10
2448                                    <1>         ;jmp    short systimer_0
2449                                    <1>       ;jmp   short systimer_19
2450 0000E203 74B6                     <1>       je     short systimer_19
2451                                    <1>
2452                                    <1> systimer_16:
2453                                    <1>       ; bh = 3
```

```
2454                                    <1>       ; timer event via real time clock interrupt
2455                                    <1>       ; interrupt/update frequency: 1 Hz (1 tick per second)
2456                                    <1>
2457 0000E205 B0B6                     <1>       mov    al, 182 ; (*) ; 18.2 * 10
2458 0000E207 FE05[E99B0100]           <1>       inc    byte [trtc] ; timer event via real time clock
2459 0000E20D EBAE                     <1>         jmp     short systimer_0
2460                                    <1>
2461                                    <1> systimer_7:
2462 0000E20F A2[64030300]             <1>       mov    [u.r0], al ; timer event number
2463                                    <1>       ;
2464                                    <1>       ; edi = address of empty timer event area
2465 0000E214 A0[B3030300]             <1>       mov    al, [u.uno]
2466 0000E219 FA                       <1>       cli    ; disable interrupts
2467 0000E21A AA                       <1>       stosb  ; process number
2468 0000E21B A0[E89B0100]             <1>       mov    al, [tcallback] ; timer callback flag
2469 0000E220 AA                       <1>       stosb  ; 1= callback method, 0= signal response byte method
2470 0000E221 A0[E99B0100]             <1>       mov    al, [trtc] ; timer interrupt type
2471 0000E226 AA                       <1>       stosb  ; 1= real time clock, 0= programmable interval timer
2472 0000E227 88D8                     <1>       mov    al, bl ; Signal return (Response) value
2473 0000E229 AA                       <1>       stosb   ; response byte
2474 0000E22A 58                       <1>       pop    eax ; (*) ; 10 or 182
2475 0000E22B 89D3                     <1>       mov    ebx, edx ; virtual address for response/signal byte
2476 0000E22D F7E1                     <1>       mul    ecx
2477                                    <1>       ; (eax = 10 * count of 18.2 Hz timer ticks)
2478                                    <1>       ; (count down step = 10)
2479 0000E22F AB                       <1>       stosd  ; count limit (reset value)
2480 0000E230 AB                       <1>       stosd  ; current count value
2481                                    <1>
2482                                    <1>       ; 19/12/2016
2483 0000E231 803D[E89B0100]00         <1>       cmp    byte [tcallback], 0 ; timer callback method ?
2484 0000E238 7604                     <1>       jna    short systimer_17 ; no
2485 0000E23A 89D8                     <1>       mov    eax, ebx ; virtual address for callback routine
2486 0000E23C EB0D                     <1>       jmp    short systimer_18
2487                                    <1>
2488                                    <1> systimer_17: ; signal response byte method
2489                                    <1>       ; ebx = virtual address
2490                                    <1>       ; [u.pgdir] = page directory's physical address
2491                                    <1>       ; 20/02/2017
2492 0000E23E FE05[EA9B0100]           <1>       inc     byte [no_page_swap] ; 1
2493                                    <1>                   ; Do not add this page to swap queue
2494                                    <1>                   ; and remove it from swap queue if it is
2495                                    <1>                   ; on the queue.
2496 0000E244 E8A780FFFF               <1>       call   get_physical_addr
2497 0000E249 721A                     <1>       jc     short systimer_8 ; 07/06/2016
2498                                    <1>       ; eax = physical address of the virtual address in user's space
2499                                    <1> systimer_18:
2500 0000E24B AB                       <1>       stosd  ; response addr (physical) or callback addr (virtual)
2501 0000E24C FE05[AB960100]           <1>       inc    byte [timer_events] ; 07/06/201
2502                                    <1>       ; 02/01/2017
2503 0000E252 0FB605[B3030300]         <1>       movzx  eax, byte [u.uno]
2504 0000E259 FE80[FF000300]           <1>       inc    byte [eax+p.timer-1]
2505                                    <1>       ;
2506 0000E25F FB                       <1>       sti    ; enable interrupts
2507 0000E260 E9A7F6FFFF               <1>       jmp    sysret
2508                                    <1>
2509                                    <1> systimer_8:
2510                                    <1>       ; 10/06/2016
2511                                    <1>       ; 07/06/2016
2512 0000E265 28C0                     <1>       sub    al, al ; 0
2513 0000E267 8847F4                   <1>       mov    [edi-12], al ; clear process number (free timer event)
2514                                    <1>       ;mov   dword [edi], eax ; 0
2515 0000E26A FB                       <1>       sti
2516 0000E26B A2[64030300]             <1>       mov    [u.r0], al ; 0
2517 0000E270 E977F6FFFF               <1>       jmp    error
2518                                    <1>
2519                                    <1> systimer_9:
2520                                    <1>       ; 10/06/2016
2521                                    <1>       ; 07/06/2016
2522 0000E275 28C0                     <1>       sub    al, al
2523 0000E277 A2[64030300]             <1>       mov    byte [u.r0], al ; 0
2524 0000E27C 3805[AB960100]           <1>       cmp     byte [timer_events], al ;  0
2525 0000E282 7631                     <1>       jna    short systimer_12
2526                                    <1>
2527                                    <1>       ; Note: ecx and edx are undefined here
2528                                    <1>       ;      (for stop timer function)
2529                                    <1>
2530 0000E284 BE[60040300]             <1>       mov    esi, timer_set  ; beginning address of timer events
2531                                    <1>                            ; setting space
2532 0000E289 A0[B3030300]             <1>       mov    al, [u.uno]
2533                                    <1>
2534 0000E28E B710                     <1>       mov    bh, 16
2535                                    <1>
2536 0000E290 08DB                     <1>       or     bl, bl
2537 0000E292 7544                     <1>       jnz    short systimer_15
2538                                    <1>
2539                                    <1>       ; clear timer event areas belong to current process
2540                                    <1>       ; (for stopping all timer events belong to current process)
2541 0000E294 FA                       <1>       cli    ; disable interrupts
2542                                    <1> systimer_10:
2543                                    <1>       ; 10/06/2016
2544                                    <1>       ; 07/06/2016
2545 0000E295 8A26                     <1>       mov    ah, [esi]
2546 0000E297 08E4                     <1>       or     ah, ah ; 0 ?
2547 0000E299 7411                     <1>       jz     short systimer_11
2548 0000E29B 38C4                     <1>       cmp    ah, al ; is the process number (owner) same ?
2549 0000E29D 750D                     <1>         jne     short systimer_11 ; no
2550                                    <1>
2551                                    <1>       ;mov   byte [esi], 0
2552 0000E29F 66C7060000               <1>       mov    word [esi], 0 ; clear
2553                                    <1>       ;mov   dword [esi+12], 0 ; clear
2554                                    <1>
2555 0000E2A4 FE0D[AB960100]           <1>       dec    byte [timer_events]
2556 0000E2AA 7409                     <1>       jz     short systimer_12
2557                                    <1>
2558                                    <1> systimer_11:
```

```
2559 0000E2AC FECF                <1>       dec    bh
2560 0000E2AE 7405                <1>       jz     short systimer_12
2561 0000E2B0 83C610              <1>       add    esi, 16
2562 0000E2B3 EBE0                <1>       jmp    short systimer_10
2563                              <1>
2564                              <1> systimer_12:
2565 0000E2B5 0FB635[B3030300]    <1>       movzx  esi, byte [u.uno]
2566 0000E2BC 08DB                <1>       or     bl, bl ; all timer events or one timer event ?
2567 0000E2BE 740C                <1>       jz     short systimer_13
2568 0000E2C0 8A9E[FF000300]      <1>       mov    bl, [esi+p.timer-1]
2569 0000E2C6 20DB                <1>       and    bl, bl ; previous number of timer events for the process
2570 0000E2C8 7408                <1>       jz     short systimer_14
2571 0000E2CA FECB                <1>       dec    bl  ; previous number of timer events for the process - 1
2572                              <1> systimer_13:
2573 0000E2CC 889E[FF000300]      <1>       mov    [esi+p.timer-1], bl ; 0 ; no timer events for process
2574                              <1> systimer_14:
2575 0000E2D2 FB                  <1>       sti    ; enable interrupts
2576 0000E2D3 E934F6FFFF          <1>       jmp    sysret
2577                              <1>
2578                              <1> systimer_15:
2579 0000E2D8 38FB                <1>       cmp    bl, bh ; 16
2580 0000E2DA 0F8707FFFFFF        <1>         ja     systimer_4      ; max. 16 timer events !
2581                              <1>       ;
2582 0000E2E0 88DA                <1>       mov    dl, bl
2583 0000E2E2 FECA                <1>       dec    dl  ; 16 -> 15 ... 1 -> 0
2584 0000E2E4 C0E204              <1>       shl    dl, 4 ; * 16
2585 0000E2E7 0FB6FA              <1>       movzx  edi, dl
2586 0000E2EA 01F7                <1>       add    edi, esi ; timer_set
2587                              <1>
2588 0000E2EC 3A07                <1>       cmp    al, [edi] ; process number
2589 0000E2EE 0F85F3FEFFFF        <1>         jne    systimer_4
2590                              <1>
2591                              <1>       ; same process ID
2592 0000E2F4 FA                  <1>       cli    ; disable interrupts
2593                              <1>       ; 10/06/2016 ; 02/01/2017
2594                              <1>       ;mov   byte [edi], 0
2595 0000E2F5 66C7070000          <1>       mov    word [edi], 0 ; clear
2596                              <1>       ;mov   dword [edi+12], 0 ; clear
2597 0000E2FA FE0D[AB960100]      <1>       dec    byte [timer_events]
2598 0000E300 EBB3                <1>       jmp    short systimer_12
2599                              <1>
2600                              <1> sysvideo: ; VIDEO DATA TRANSFER FUNCTIONS
2601                              <1>       ; 02/03/2021
2602                              <1>       ; 28/02/2021
2603                              <1>       ; 27/02/2021
2604                              <1>       ; 26/02/2021
2605                              <1>       ; 25/02/2021
2606                              <1>       ; 21/02/2021, 22/02/2021, 23/02/2021
2607                              <1>       ; 15/02/2021, 16/02/2021, 18/02/2021
2608                              <1>       ; 10/02/2021, 11/02/2021, 12/02/2021
2609                              <1>       ; 07/02/2021, 08/02/2021
2610                              <1>       ; 01/02/2021, 02/02/2021, 05/02/2021
2611                              <1>       ; 29/01/2021, 30/01/2021, 31/01/2021
2612                              <1>       ; 23/01/2021, 24/01/2021, 28/01/2021
2613                              <1>       ; 18/01/2021, 19/01/2021, 22/01/2021
2614                              <1>       ; 04/01/2021, 10/01/2021, 11/01/2021
2615                              <1>       ; 01/01/2021, 02/01/2021, 03/01/2021
2616                              <1>       ; 28/12/2020, 29/12/2020, 30/12/2020
2617                              <1>       ; 25/12/2020, 26/12/2020
2618                              <1>       ; 21/12/2020, 23/12/2020
2619                              <1>       ; 12/12/2020, 14/12/2020
2620                              <1>       ; 10/12/2020, 11/12/2020
2621                              <1>       ; 03/12/2020, 04/12/2020
2622                              <1>       ; 22/11/2020, 23/11/2020
2623                              <1>       ; 21/11/2020 (TRDOS 386 v2.0.3)
2624                              <1>       ; 12/05/2017
2625                              <1>       ; 11/07/2016
2626                              <1>       ; 13/06/2016
2627                              <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
2628                              <1>       ;
2629                              <1>       ; VIDEO DATA TRANSFER FUNCTIONS:
2630                              <1>       ;
2631                              <1>       ; Inputs:
2632                              <1>       ;            ; 07/02/2021
2633                              <1>       ;     BH = 0 = VIDEO BIOS Mode 3, tty/text mode data transfers
2634                              <1>       ;          BL =
2635                              <1>       ;            Bits 0&1, Transfer direction
2636                              <1>       ;                  0 - System to system
2637                              <1>       ;                  1 - User to system
2638                              <1>       ;                  2 - System to user
2639                              <1>       ;                  3 - Exhange (Swap) - 28/01/2021
2640                              <1>       ;            Bits 2, Transfer Type
2641                              <1>       ;                  0 - Display page (complete) transfer
2642                              <1>       ;                  1 - Display page window (col,row) transfer
2643                              <1>       ;            ; 28/01/2021
2644                              <1>       ;            Bits 3..7 - Reserved, undefined (must be 0)
2645                              <1>       ;             ; 28/01/2021
2646                              <1>       ;          /// BL = 0 -> System to system (display page) transfer
2647                              <1>       ;             CL = Source page (0FFh = current video page)
2648                              <1>       ;             DL = Destination page (0FFh = current video page)
2649                              <1>       ;             (Note: Nothing to do if src & dest are same page)
2650                              <1>       ;          /// BL = 1&2 -> user to system & system to user transfer
2651                              <1>       ;             ECX = User's buffer address
2652                              <1>       ;             DL = Video page (0FFh = current video page)
2653                              <1>       ;          /// BL = 3 -> exchange (swap) display page ; 28/01/2021
2654                              <1>       ;             ECX = User's buffer address
2655                              <1>       ;             DL = Video page (0FFh = current video page)
2656                              <1>       ;             EDI = Swap address in user's memory (must be > 0)
2657                              <1>       ;          /// BL = 5&6&7 -> user to system, system to user transfer
2658                              <1>       ;            (system window is in current/active display page)
2659                              <1>       ;             ESI = User's buffer address
2660                              <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2661                              <1>       ;             ECX High 16 bits = Top row (Y1 position)
2662                              <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
2663                              <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
```

```
2664                    <1>         ;               If BL = 5 or BL bit 0 & bit 1 are 1 ; 28/01/2021
2665                    <1>       ;             EDI = Swap address (in user's memory space)
2666                    <1>       ;             (If swap address > 0, previous content of the window
2667                    <1>       ;             will be saved into swap area in user's memory space)
2668                    <1>       ;           /// BL = 4 -> system to system transfer
2669                    <1>       ;             ESI = System's source buffer (video page) address
2670                    <1>       ;             ECX Low 16 bits = Top left column (X1 position)
2671                    <1>       ;             ECX High 16 bits = Top row (Y1 position)
2672                    <1>       ;             EDX Low 16 bits = Bottom right column (X2 position)
2673                    <1>       ;             EDX High 16 bits = Bottom row (Y2 position)
2674                    <1>       ;             EDI = System's destination buffer (video page) addr
2675                    <1>       ;
2676                    <1>       ;             ; 06/02/2021
2677                    <1>       ;             ; 05/02/2021
2678                    <1>       ;             ; 01/02/2021, 02/02/2021
2679                    <1>       ;             ; 30/01/2021, 31/02/2021
2680                    <1>       ;             ; 29/01/2021 (major modification)
2681                    <1>       ;             ; 23/11/2020 (major modification)
2682                    <1>       ;             ; 22/11/2020 (bugfixes and extensions)
2683                    <1>       ;     BH = 1 = VGA Graphics (0A0000h) data transfers
2684                    <1>       ;             BL bit 7
2685                    <1>       ;                resolution (screen width) option
2686                    <1>       ;                0 = 320 pixels
2687                    <1>       ;                1 = 640 pixels
2688                    <1>       ;             .. followings are same with SVGA transfer function
2689                    <1>       ;             BL bit 6
2690                    <1>       ;                direction option
2691                    <1>       ;                0 = user to system (video memory)
2692                    <1>       ;                1 = system to user
2693                    <1>       ;             BL bit 5
2694                    <1>       ;                masked/direct (non-masked) operations
2695                    <1>       ;                1 = masked, 0 = non-masked (direct)
2696                    <1>       ;             BL bit 4
2697                    <1>       ;                page/window option
2698                    <1>       ;                1 = window, 0 = display page (screen)
2699                    <1>       ;             BL bit 0 to 3 (pixel operation types)
2700                    <1>       ;                0 = Copy pixels (colors) ((mask color))
2701                    <1>       ;                1 = Change (New, Fill) color
2702                    <1>       ;                2 = Add color
2703                    <1>       ;                3 = Sub color
2704                    <1>       ;                4 = OR color
2705                    <1>       ;                5 = AND color
2706                    <1>       ;                6 = XOR color
2707                    <1>       ;                7 = NOT color
2708                    <1>       ;                8 = NEG color
2709                    <1>       ;                9 = INC color
2710                    <1>       ;               10 = DEC color
2711                    <1>       ;               11 = Mix (Average) colors
2712                    <1>       ;               12 = Replace pixel colors
2713                    <1>       ;               13 = Copy pixel block(s)
2714                    <1>       ;               14 = Write line(s)
2715                    <1>       ;               15 = Write character (font)
2716                    <1>       ;
2717                    <1>       ;          Input Registers for pixel operations:
2718                    <1>       ;              Same with LFB data transfer function below
2719                    <1>       ;
2720                    <1>       ;             ; 25/02/2021
2721                    <1>       ;             ; 05/02/2021, 06/02/2021
2722                    <1>       ;             ; 01/02/2021, 02/02/2021
2723                    <1>       ;             ; 30/01/2021, 31/02/2021
2724                    <1>       ;             ; 29/01/2021 (major modification)
2725                    <1>       ;             ; 23/11/2020 (major modification)
2726                    <1>       ;             ; 22/11/2020 (bugfixes and extensions)
2727                    <1>       ;       BH = 2 = Super VGA, LINEAR FRAME BUFFER data transfers
2728                    <1>       ;             BL bit 7
2729                    <1>       ;                unused (invalid), must be 0
2730                    <1>       ;             BL bit 6
2731                    <1>       ;                direction option
2732                    <1>       ;                0 = user to system (video memory)
2733                    <1>       ;                1 = system to user
2734                    <1>       ;             BL bit 5
2735                    <1>       ;                masked/direct (non-masked) operations
2736                    <1>       ;                1 = masked, 0 = non-masked (direct)
2737                    <1>       ;             BL bit 4
2738                    <1>       ;                page/window option
2739                    <1>       ;                1 = window, 0 = display page (screen)
2740                    <1>       ;             BL bit 0 to 3 (pixel operation types)
2741                    <1>       ;                0 = Copy pixels (colors) ((mask color))
2742                    <1>       ;                1 = Change (New, Fill) color
2743                    <1>       ;                2 = Add color
2744                    <1>       ;                3 = Sub color
2745                    <1>       ;                4 = OR color
2746                    <1>       ;                5 = AND color
2747                    <1>       ;                6 = XOR color
2748                    <1>       ;                7 = NOT color
2749                    <1>       ;                8 = NEG color
2750                    <1>       ;                9 = INC color
2751                    <1>       ;               10 = DEC color
2752                    <1>       ;               11 = Mix (Average) colors
2753                    <1>       ;               12 = Replace pixel colors
2754                    <1>       ;               13 = Copy pixel block(s)
2755                    <1>       ;               14 = Write line(s)
2756                    <1>       ;               15 = Write character (font)
2757                    <1>       ;
2758                    <1>       ;          Note: If HW of EBX > 0, it is VESA VBE mode number
2759                    <1>       ;               otherwise, function will be applied
2760                    <1>       ;               to current (VESA VBE) video mode.
2761                    <1>       ;
2762                    <1>       ;          Input Registers for pixel operations:
2763                    <1>       ;             -- user to system & system to system --
2764                    <1>       ;             -- (BL = 0 to 0Fh) -- non-masked, screen --
2765                    <1>       ;             -- (BL = 10h to 1Fh) -- non-masked, window --
2766                    <1>       ;             -- (BL = 20h to 2Fh) -- masked, screen --
2767                    <1>       ;             -- (BL = 30h to 3Fh) -- masked, window --
2768                    <1>       ;             (*) window, (**) masked (***) sys to sys
```

```
2769                    <1>     ;           for BL bit 0 to 3
2770                    <1>     ;           00h: COPY PIXELS
2771                    <1>     ;             If BL bit 4 = 0 ; 21/02/2021
2772                    <1>     ;                full screen copy
2773                    <1>     ;                ECX & EDX will not be used
2774                    <1>     ;               (user buffer must fit to display page)
2775                    <1>     ;             If BL bit 4 = 1 ; 21/02/2021
2776                    <1>     ;               ECX = start position (row, column) (*)
2777                    <1>     ;                    (HW = row, CX = column)
2778                    <1>     ;               EDX = size (rows, colums) (*)
2779                    <1>     ;                    (HW = rows, DX = columns)
2780                    <1>     ;                     (0 -> invalid)
2781                    <1>     ;                     (1 -> horizontal or vertical line)
2782                    <1>     ;               ESI = user's buffer address
2783                    <1>     ;               EDI = mask color (**) ; 25/02/2021
2784                    <1>     ;                    (this color will be excluded)
2785                    <1>     ;           01h: CHANGE PIXEL COLORS
2786                    <1>     ;           02h: ADD PIXEL COLORS
2787                    <1>     ;           03h: SUB PIXEL COLORS
2788                    <1>     ;           04h: OR PIXEL COLORS
2789                    <1>     ;           05h: AND PIXEL COLORS
2790                    <1>     ;           06h: XOR PIXEL COLORS
2791                    <1>     ;           0Bh: MIX PIXEL COLORS
2792                    <1>     ;                CL = color (8 bit, 256 colors)
2793                    <1>     ;               ECX = color (16 bit and true colors)
2794                    <1>     ;               EDX = start position (row, column) (*)
2795                    <1>     ;                   (HW = row, DX = column)
2796                    <1>     ;               ESI = size (rows, colums) (*)
2797                    <1>     ;                    (HW = rows, SI = columns)
2798                    <1>     ;               EDI = mask color (**) ; 25/02/2021
2799                    <1>     ;                    (this color will be excluded)
2800                    <1>     ;           07h: NOT PIXEL COLORS
2801                    <1>     ;           08h: NEG PIXEL COLORS
2802                    <1>     ;           09h: INC PIXEL COLORS
2803                    <1>     ;           0Ah: DEC PIXEL COLORS
2804                    <1>     ;               ECX = start position (row, column) (*)
2805                    <1>     ;                    (HW = row, CX = column)
2806                    <1>     ;               EDX = size (rows, colums) (*)
2807                    <1>     ;                    (HW = rows, DX = columns)
2808                    <1>     ;                     (0 -> invalid)
2809                    <1>     ;                     (1 -> horizontal or vertical line)
2810                    <1>     ;               EDI = mask color (**) ; 25/02/2021
2811                    <1>     ;                    (this color will be excluded)
2812                    <1>     ;           0Ch: REPLACE PIXEL COLORS
2813                    <1>     ;                CL = current color (8 bit, 256 colors)
2814                    <1>     ;               ECX = current color (16 bit and true colors)
2815                    <1>     ;                DL = new color (8 bit, 256 colors)
2816                    <1>     ;               EDX = new color (16 bit and true colors)
2817                    <1>     ;               ESI = start position (row, column) (*)
2818                    <1>     ;                    (HW = row, SI = column)
2819                    <1>     ;               EDI = size (rows, colums) (*)
2820                    <1>     ;                    (HW = rows, DI = columns)
2821                    <1>     ;           0Dh: COPY PIXEL BLOCK(S) -full screen-
2822                    <1>     ;              -If BL bit 5 is 0-
2823                    <1>     ;               ECX = start position (row, column) (*)
2824                    <1>     ;                    (HW = row, CX = column)
2825                    <1>     ;               EDX = size (rows, colums) (*)
2826                    <1>     ;                    (HW = rows, DX = columns)
2827                    <1>     ;                     (0 -> invalid)
2828                    <1>     ;                     (1 -> horizontal or vertical line)
2829                    <1>     ;               ESI = destination (row, column) (***)
2830                    <1>     ;              -If BL bit 5 is 1-
2831                    <1>     ;                   CL = color (8 bit, 256 colors)
2832                    <1>     ;                 ECX = color (16 bit and true colors)
2833                    <1>     ;                 EDX = count of blocks (not bytes)
2834                    <1>     ;                      (limit: 2048 blocks)
2835                    <1>     ;                 ESI = user's buffer address
2836                    <1>     ;                    contains 64 bits block data
2837                    <1>     ;                    BLOCK ADDRESS - (row, col), dword
2838                    <1>     ;                    (first 32 bits)
2839                    <1>     ;                    BLOCK SIZE - (rows, cols), dword
2840                    <1>     ;                    (second 32 bits)
2841                    <1>     ;           ; 10/02/2021
2842                    <1>     ;           0Eh: WRITE LINE(s) -full screen-
2843                    <1>     ;              -If BL bit 5 is 0-
2844                    <1>     ;               CL = color (8 bit, 256 colors)
2845                    <1>     ;              ECX = color (16 bit and true colors)
2846                    <1>     ;               DX = low 12 bits - size (length)
2847                    <1>     ;                   high 4 bits - direction or type
2848                    <1>     ;                       0 - Horizontal line
2849                    <1>     ;                       1 - Vertical line
2850                    <1>     ;                     > 1 - undefined, invalid
2851                    <1>     ;              ESI = start position (row, column)
2852                    <1>     ;                   (HW = row, SI = column)
2853                    <1>     ;              -If BL bit 5 is 1-
2854                    <1>     ;                CL = color (8 bit, 256 colors)
2855                    <1>     ;               ECX = color (16 bit and true colors)
2856                    <1>     ;                DX = number of lines (in user buffer)
2857                    <1>     ;                     (limit: 2048 lines)
2858                    <1>     ;               ESI = user's buffer
2859                    <1>     ;                      contains 64 bit data for lines
2860                    <1>     ;                     START POINT: 32 bit (row, col)
2861                    <1>     ;                     LENGTH: 32 bit
2862                    <1>     ;                     high 16 bits - 0
2863                    <1>     ;                     bit 0-11 - length
2864                    <1>     ;                     bit 12-15 - type (length)
2865                    <1>     ;           0Fh: WRITE CHARACTER (FONT)
2866                    <1>     ;                CL = char's color (8 bit, 256 colors)
2867                    <1>     ;               ECX = char's color (16 bit and true colors)
2868                    <1>     ;                DL = Character's ASCII code
2869                    <1>     ;                DH bit 0 -> font height
2870                    <1>     ;                    0 -> 8x16 character font
2871                    <1>     ;                    1 -> 8x8 character font
2872                    <1>     ;                DH bit 1 & 2 -> scale
2873                    <1>     ;                    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
                                ;                      DL = 2 -> 16x16 (width: 2 bytes)
                                ;                      DL = 3 -> 16x32
                                ;                      DL = 4 -> 24x24 (width: 3 bytes)
                                ;                      DL = 5 -> 24x48
                                ;                      DL = 6 -> 32x32 (width: 4 bytes)
                                ;                      DL = 7 -> 32x64
                                ;                      DL > 7 -> invalid (unused)
                                ;                   EDI = user's font buffer address
                                ;                      (NOTE: byte order is as row0,row1,row2..)
                                ;                ESI = start position (row, column) (*)
                                ;                       (HW = row, SI = column)
                                ;
                                ;             -- system to user --
                                ;                BL (bit 0 to 7)
                                ;             40h: COPY PIXELS (full screen, display page)
                                ;                 EDI = user's buffer address
                                ;             41h: COPY PIXELS (window)
                                ;                 ECX = start position (row, column) (*)
                                ;                      (HW = row, CX = column)
                                ;                 EDX = size (rows, colums) (*)
                                ;                      (HW = rows, DX = columns)
                                ;                        (<=1 -> horizontal or vertical line)
                                ;                 EDI = user's buffer address
                                ;
                                ;             Example: (29/01/2021)
                                ;                 ecx = 00400064h (start at row 64, column 100)
                                ;                 edx = 00320048h (size: 50 rows, 72 columns)
                                ;                               (end at row 114, column 172)
                                ;                 If video memory starts at 0A0000h
                                ;                 and if resolution is 320x200 (256 colors) ..
                                ;                 window start offset: (64*320)+100 = 20580
                                ;                 window size: 16072 bytes (pixels)
                                ;                 window end offset: 20580+16072 = 36652
                                ;                 window start address: 0A0000h+564h = 0A5064h
                                ; Outputs:
                                ;     EAX = transfer/byte count
                                ;
                                ;     NOTE: If the source or destination address passes out of
                                ;     video pages (display memory limits), data will not be transferred
                                ;     and EAX will return as 0.
                                ;
                                ; 08/02/2021
                                ; 07/02/2021
                                ; 04/01/2021
                                ; PIXEL READ/WRITE (in current/active video mode)
                                ;
                                ;     BH = 3 = Read/Write pixel(s) -for all graphics modes-
                                ;          BL =
                                ;            0 = Read pixel
                                ;            1 = Write pixel
                                ;            2 = swap pixel colors
                                ;            3 = mix pixel colors
                                ;          29/01/2021
                                ;            4 = read pixels from user defined positions
                                ;            5 = write single color pixels to user defined positions
                                ;            6 = write multi color pixels to user defined positions
                                ;
                                ;           > 6 = invalid/unimplemented
                                ;
                                ;          .. for BL = 0 to 5
                                ;          CL = color for writing pixel(s) or
                                ;          ECX = color for writing pixel(s) in true color modes
                                ;
                                ;          EDX = Offset from start of video memory (0A0000h)
                                ;                or start of linear frame buffer
                                ;
                                ;          07/02/2021
                                ;          .. for BL = 4
                                ;          EDI = user's destination buffer address for pixel colors
                                ;          29/01/2021
                                ;          .. for BL = 4 & 5
                                ;          ESI = user's source buffer address for BL = 4 & 5
                                ;          (buffer contains dword offset positions for pixels)
                                ;          EDX = number of pixels
                                ;          .. for BL = 6
                                ;          ESI = user's buffer address for BL = 6
                                ;          (buffer contains dword offset position and dword color
                                ;           value for each pixel)
                                ;          EDX = number of pixels
                                ;
                                ; Note:
                                ;     Pixel read/write will be performed in current video mode.
                                ;     If [CRT_MODE] < 0FFh, 0A0000h will be used
                                ;        as video memory and limit will be 65536
                                ;        (new/mix pixel color will be in CL)
                                ;     if [CRT_MODE] = 0FFh (VESA VBE video mode)
                                ;        LFB base address will be used as video memory
                                ;        and limit will be video page size
                                ;        (new/mix pixel color will be in CL)
                                ;
                                ; Outputs:
                                ;     EAX = pixel color (according to BL and ECX input)
                                ;     EAX = 0 (pixel color is 0 or there is an error)
                                ;          (BL will return as 0FFh if there is an error)
                                ;     ; 29/01/2021
                                ;     EAX = number of pixels (for BL input = 4&5&6)
                                ;
```

```
2979                    <1>     ; DIRECT (STANDARD VGA/CGA) DISPLAY MEMORY ACCESS FUNCTIONS:
2980                    <1>     ;
2981                    <1>     ;     BH = 4 = CGA direct video memory (0B8000h, 32K) access
2982                    <1>     ;            Page directory & page tables of the user's
2983                    <1>     ;            program will be updated to direct access to
2984                    <1>     ;            0B8000h (32K) video (CGA, color) memory; if
2985                    <1>     ;            there is not a permission conflict or lock!
2986                    <1>     ;              (User's program/process will have permission to
2987                    <1>     ;            access locked display memory if the owner is
2988                    <1>     ;            it's parent.)
2989                    <1>       ;
2990                    <1>     ;         Screen width = 320
2991                    <1>     ;
2992                    <1>     ;     BH = 5 = VGA direct video memory (0A0000h, 64K) access
2993                    <1>     ;            Page directory & page tables of the user's
2994                    <1>     ;            program will be updated to direct access to
2995                    <1>     ;            0A0000h (64K) video (VGA) memory; if there is not
2996                    <1>     ;            a permission conflict or lock!
2997                    <1>     ;              (User's program/process will have permission to
2998                    <1>     ;            access locked display memory if the owner is
2999                    <1>     ;            it's parent.)
3000                    <1>     ;
3001                    <1>     ;         ; 23/11/2020
3002                    <1>     ;         Screen width options = 320, 640, 800
3003                    <1>     ;
3004                    <1>     ; Outputs:
3005                    <1>     ;     EAX = Display memory address for direct access
3006                    <1>     ;           0A0000h for VGA, 0B8000h for CGA
3007                    <1>     ;     (Display memory size: 32K for CGA, 64K for VGA)
3008                    <1>     ;     EAX = 0 if display page access permission has been denied.
3009                    <1>     ;           (Locked!)
3010                    <1>     ;
3011                    <1>     ; LINEAR FRAME BUFFER ACCESS FUNCTIONS:
3012                    <1>     ;
3013                    <1>     ;     BH = 6 = Linear Frame Buffer direct video memory access
3014                    <1>     ;
3015                    <1>     ;            Page directory & page tables of the user's
3016                    <1>     ;            program will be updated to direct access to
3017                    <1>     ;            the configured LFB (Linear Frame Buffer) address,
3018                    <1>     ;            if there is not a permission conflict or lock!
3019                    <1>     ;              (User's program/process will have permission to
3020                    <1>     ;            access locked display memory if the owner is
3021                    <1>     ;            it's parent.)
3022                    <1>     ;
3023                    <1>     ;        ; 10/12/2020
3024                    <1>     ;        BL = 0FFh -> Direct LFB access for current video mode
3025                    <1>     ;        BL = XXh < 0FFh -> Direct LFB access
3026                    <1>     ;                          for VESA video mode 1XXh
3027                    <1>     ;
3028                    <1>     ;            Return: EAX = Linear Frame Buffer address
3029                    <1>     ;                   (EAX = 0 -> error)
3030                    <1>     ;                 If EAX > 0
3031                    <1>     ;                   EDX = Frame Buffer Size in bytes
3032                    <1>     ;                     BH = Requested Video Mode - 100h
3033                    <1>     ;                          (VESA VBE video modes)
3034                    <1>     ;                     BL = bits per pixel
3035                    <1>     ;                         8 = 256 colors, 8
3036                    <1>     ;                         16 = 65536 colors, 5-6(G)-5
3037                    <1>     ;                         24 = RGB, 16M colors, 8-8-8
3038                    <1>     ;                         32 = RGB + alpha bytes, 8-8-8-8
3039                    <1>     ;                 If BH = 0FFh
3040                    <1>     ;                    BL = VGA/CGA video mode (also EAX = 0)
3041                    <1>     ;
3042                    <1>     ;            ** Function will return with EAX = 0 if the mode
3043                    <1>     ;            is not a valid VESA VBE video mode as 1??h **
3044                    <1>     ;
3045                    <1>     ;            ECX = Pixel resolution
3046                    <1>     ;                  CX = Width (320, 640, 800, 1024, 1366, 1920)
3047                    <1>     ;                  High 16 bits of ECX = Height
3048                    <1>     ;
3049                    <1>     ;     23/11/2020
3050                    <1>     ; ***  GET VIDEO MODE & LINEAR FRAME BUFFER INFO
3051                    <1>     ;     (This function -7- also is used for VGA and CGA modes)
3052                    <1>     ;
3053                    <1>     ;     BH = 7 = Get Linear Frame Buffer info
3054                    <1>     ;
3055                    <1>     ;        ; 22/01/2021
3056                    <1>     ;        ; 10/12/2020
3057                    <1>     ;        BL = any -not used- (22/01/2021)
3058                    <1>     ;
3059                    <1>     ;            Return:
3060                    <1>     ;            EAX = Frame Buffer Address (0 = is not in use)
3061                    <1>     ;            EDX = Frame Buffer Size in bytes
3062                    <1>     ;             BH = Current Video Mode - 100h  ; 22/01/2021
3063                    <1>     ;                 (VESA VBE video modes)
3064                    <1>     ;             BL = bits per pixel
3065                    <1>     ;                   8 = 256 colors, 8
3066                    <1>     ;                   16 = 65536 colors, 5-6(G)-5
3067                    <1>     ;                   24 = RGB, 16M colors, 8-8-8
3068                    <1>     ;                   32 = RGB + alpha bytes, 8-8-8-8
3069                    <1>     ;            If BH = 0FFh
3070                    <1>     ;               BL = VGA/CGA video mode (also EAX = 0)
3071                    <1>     ;
3072                    <1>     ;            Note:
3073                    <1>     ;            Alpha byte will be used as virtual color index.
3074                    <1>     ;            (32 bit pixel colors.. byte 0,1,2 rgb and 3 alpha)
3075                    <1>     ;
3076                    <1>     ;            ** Function will return with EAX = 0 if the mode
3077                    <1>     ;            is not a valid VESA VBE video mode as 1??h **
3078                    <1>     ;
3079                    <1>     ;            ECX = Pixel resolution
3080                    <1>     ;                  CX = Width (320, 640, 800, 1024, 1366, 1920)
3081                    <1>     ;                  High 16 bits of ECX = Height
3082                    <1>     ;
3083                    <1>     ;     NOTE: Each process will have it's own frame buffer
```

```
3084                    <1>     ;           address and resolution parameters in 'u' area.
3085                    <1>     ;           Then, if the current frame buffer & resolution
3086                    <1>     ;           is different, frame buffer r/w functions
3087                    <1>     ;           will use scale factor to convert process's
3088                    <1>       ;             pixel coordinates to actual screen coordinates.
3089                    <1>     ;           resolution -> dimensional scale
3090                    <1>     ;           color size -> color scale
3091                    <1>     ;         * RGB (TRUE) colors to 256 colors conversion:
3092                    <1>       ;             TRUE Colors -> 8,8,8 (R,G,B; byte 0 is R)
3093                    <1>     ;           256 colors -> 2,2,2,2 (R,G,B,L; bit 0&1 is R)
3094                    <1>     ;              bit 6&7 -> luminosity base level (0,1,2,3)
3095                    <1>     ;              bit 4&5 -> blue level (0,1,2,3)
3096                    <1>     ;              bit 2%3 -> green level (0,1,2,3)
3097                    <1>     ;              bit 0&1 -> red level (0,1,2,3)
3098                    <1>     ;           Example: total red level : luminosity + red level
3099                    <1>     ;           Luminosity base level: 0 -> 16
3100                    <1>     ;                                  1 -> 32
3101                    <1>     ;                                  2 -> 64
3102                    <1>     ;                                  3 -> 128
3103                    <1>     ;           Color level:
3104                    <1>     ;                                 0 -> 0
3105                    <1>     ;                                 1 -> luminosity level
3106                    <1>     ;                                 2 -> luminosity level + 64
3107                    <1>     ;                                 3 -> 255
3108                    <1>     ;         Luminosity base level = min (R,G,B)
3109                    <1>     ;                 if it is <16, it will be set to 16
3110                    <1>     ;         Color levels: Color values are fixed to (nearest)
3111                    <1>     ;             one of all possible set level (step) values
3112                    <1>     ;             (according to luminosity base level); then
3113                    <1>     ;             color levels are set to R-L, G-L, B-L.
3114                    <1>     ;       For example: If luminosity base level is 32
3115                    <1>     ;              all possible set values are 0, 32, 96, 255.
3116                    <1>     ;
3117                    <1>     ;         * RGB (TRUE) colors to 16 colors conversion:
3118                    <1>     ;         16 colors: R,B,G,L bits (4 bits)
3119                    <1>     ;               If any one of R,G,B >= 128 L = 1
3120                    <1>     ;               If max. value of (R,G,B) >= 32, it is 1
3121                    <1>     ;                  else all color bits (R&G&B&L) are 0
3122                    <1>     ;               If the second value >= max. value / 2
3123                    <1>     ;                  it is 1
3124                    <1>     ;               If third value value >= max. value / 2
3125                    <1>     ;                  it is 1
3126                    <1>     ;         Example: R = 132, G = 64, B = 78
3127                    <1>     ;                 L = 1, R = 1
3128                    <1>     ;                 G < 66 --> G = 0
3129                    <1>     ;                 B >= 66 --> B = 1
3130                    <1>     ;
3131                    <1>     ; 10/12/2020
3132                    <1>     ; SET VIDEO MODE (& RETURN LFB INFO for VESA VBE VIDEO MODES)
3133                    <1>     ;
3134                    <1>     ;     BH = 8 = Set Video Mode
3135                    <1>     ;
3136                    <1>     ;           BL = Requested Video Mode (method)
3137                    <1>     ;           If BL = 0FFh
3138                    <1>     ;              CX = VESA VBE Video Mode
3139                    <1>     ;              ; 11/12/2020
3140                    <1>     ;              If EDX > 0 -> LFB INFO (user) buffer addr
3141                    <1>     ;           If BL < 0FFh, it is VGA/CGA video mode and
3142                    <1>     ;                CX & EDX will not be used
3143                    <1>     ;
3144                    <1>     ;         NOTE: The last VESA VBE video mode is 11Bh but
3145                    <1>     ;         TRDOS 386 will permit to set video mode upto 11Fh.
3146                    <1>     ;         Above 11Fh, from 140h to 1FEh, it will be accepted
3147                    <1>     ;         as Bochs/Plex86 emulator video mode and it will be
3148                    <1>     ;         used only if [vbe3] = 2 and detected
3149                    <1>     ;         video bios is BOCHS/PLEX86/QEMU/VIRTUALBOX vbios.
3150                    <1>     ;
3151                    <1>     ; Outputs:
3152                    <1>     ;     EAX = Requested (Proper) video mode number + 1
3153                    <1>     ;           ("dec eax" by user will give requested video mode),
3154                    <1>     ;
3155                    <1>     ;     If BL input is 0FFh
3156                    <1>     ;        EDX = LFBINFO table/structure (in user's buffer addr)
3157                    <1>     ;        EDX = 0 -> Invalid LFBINFO (do not use it)
3158                    <1>     ;
3159                    <1>     ;     EAX = 0 -> Error (but EDX will not be changed)
3160                    <1>     ;
3161                    <1>     ; 03/12/2020
3162                    <1>     ; VESA VBE3 VIDEO BIOS (32 bit) PROTECTED MODE INTERFACE SETTINGS
3163                    <1>     ;
3164                    <1>     ;     BH = 9 = set/get VBE3 Protected Mode Interface parameters
3165                    <1>     ;
3166                    <1>     ;           BL = 0 - Disable protected mode interface
3167                    <1>     ;                  ([pmi32] = 0)
3168                    <1>     ;                  Return: AL = 1
3169                    <1>     ;           BL = 1 - Enable protected mode Interface
3170                    <1>     ;                  ([pmi32] = 1)
3171                    <1>     ;                  Return: AL = 2
3172                    <1>     ;           BL = 2 - Get protected mode interface status
3173                    <1>     ;                  Return: AL = [pmi32] + 1 (AL = 1 or 2)
3174                    <1>     ;
3175                    <1>     ;           If [vbe3] <> 3 --> AL = 0
3176                    <1>     ;
3177                    <1>     ;           ; 17/01/2021
3178                    <1>     ;           BL = 3 - Disable/Cancel restore permission to user
3179                    <1>     ;                  Return: AL = 1 (if disabled) or 0
3180                    <1>     ;           BL = 4 - Enable/Give restore permission to user
3181                    <1>     ;                  Return: AL = 2 (if enabled) or 0
3182                    <1>     ;           BL = 5 - Get video state save/restore status
3183                    <1>     ;                   (permission status)
3184                    <1>     ;                   Return: AL = Status (enabled = 1)
3185                    <1>     ;                        ; 22/01/2021
3186                    <1>     ;                        AH = state options ([srvso])
3187                    <1>     ;           BL = 6 - Return VESA VBE number/status
3188                    <1>     ;                   Return: AX = status
```

```
3189                    <1>     ;                       if AH =  2, AL > 0 : Emulator
3190                    <1>     ;                         AH =  3, VESA VBE3 video bios
3191                    <1>     ;           ; 28/02/2021
3192                    <1>     ;           BL = 7 - Set true color mode as 32bpp (default)
3193                    <1>     ;                 Return: AX = 32 (if VBE3)
3194                    <1>     ;           NOTE: Initial/default value is 32bpp for vbe3.
3195                    <1>     ;                 Return: AX = 0 -> error
3196                    <1>     ;           BL = 8 - Set true color mode as 24bpp (default)
3197                    <1>     ;                 Return: AX = 24
3198                    <1>     ;                 ;Return: AX = 0 -> error
3199                    <1>     ;           BL = 9 - Return default/current true color mode
3200                    <1>     ;                 Return: AX = 32 (32 bpp)
3201                    <1>     ;                 Return: AX = 24 (24 bpp)
3202                    <1>     ;                 Return: AX = 0 -> error (not VESA bios)
3203                    <1>     ;
3204                    <1>     ;           BL > 9 : not implemented (28/02/2021)
3205                    <1>     ;
3206                    <1>     ;     ; 19/01/2021 ([u.uid] check)
3207                    <1>     ;     Note: Enabling/Disabling are done by root ([u.uid] = 0)
3208                    <1>     ;           while [multi_tasking] = 0.
3209                    <1>     ;
3210                    <1>     ; 23/12/2020
3211                    <1>     ; VIDEO MEMORY MAPPING:
3212                    <1>     ;     BH = 10 = Map video memory to user's buffer
3213                    <1>     ;
3214                    <1>     ;        BL = 0 : CGA memory (0B8000h) map (32K)
3215                    <1>     ;        BL = 1 : VGA memory (0A0000h) map (64K)
3216                    <1>     ;        BL = 2 : SVGA memory (LFB) map to user's buffer
3217                    <1>     ;
3218                    <1>     ;     ECX = User's buffer addr (low 12 bits will be cleared)
3219                    <1>     ;     EDX = Buffer size in bytes (if BL = 2)
3220                    <1>     ;           (will be trimmed if LFB size < EDX)
3221                    <1>     ; Return:
3222                    <1>     ;   EAX = physical address of video memory (buffer)
3223                    <1>     ;     EBX = mapped (actual) size of video memory (bytes)
3224                    <1>     ;     ECX = virtual start address of user's video buffer
3225                    <1>     ;     EDX is same with EDX input
3226                    <1>     ;
3227                    <1>     ;     (Note: Memory page boundaries will be applied
3228                    <1>     ;      to buffer size and buff start addr by rounding down.
3229                    <1>     ;      Rounded size & address values must not be zero.)
3230                    <1>     ;     -Normally, it is expected to request mapping by using
3231                    <1>     ;      correct buffer size of current or desired video mode-
3232                    <1>     ;
3233                    <1>     ;     EAX = 0 -> error ! memory can not mapped to user
3234                    <1>     ;
3235                    <1>     ; 04/01/2021
3236                    <1>     ; SET/READ COLOR PALETTE (set/read DAC color registers)
3237                    <1>     ;     ((256 colors (8bpp) VGA/CGA video hardware feature))
3238                    <1>     ;
3239                    <1>     ;     BH = 11 = Set/Read DAC color registers
3240                    <1>     ;
3241                    <1>     ;        (BL<4 Original method for std VGA video hardware)
3242                    <1>     ;        BL = 0 : Read all DAC color registers (256 colors)
3243                    <1>     ;            (6 bit colors, in RGB order)
3244                    <1>     ;        BL = 1 : Set all DAC color registers (256 colors)
3245                    <1>     ;            (6 bit colors, in RGB order)
3246                    <1>     ;        BL = 2 : Read single DAC color register
3247                    <1>     ;            (6 bit color, in RGB order)
3248                    <1>     ;           ((EAX will return with color value))
3249                    <1>     ;           CL = DAC color register (index)
3250                    <1>     ;        BL = 3 : Set/Write single DAC color register
3251                    <1>     ;            (6 bit color, in RGB order, bit 6&7 are 0)
3252                    <1>     ;           ECX byte 0 - DAC color register
3253                    <1>     ;           ECX byte 1 - Red (6 bit)
3254                    <1>     ;           ECX byte 2 - Green (6 bit)
3255                    <1>     ;           ECX byte 3 - Blue (6 bit)
3256                    <1>     ;        (BL>3 Alternative method for BMP files etc.)
3257                    <1>     ;        BL = 4 : Read all DAC color registers (256 colors)
3258                    <1>     ;            (8 bit colors, in BGR order, bit 0&1 is 0)
3259                    <1>     ;        BL = 5 : Set all DAC color registers (256 colors)
3260                    <1>     ;            (8 bit colors, in BGR order, bit 0&1 is 0)
3261                    <1>     ;        BL = 6 : Read single DAC color register
3262                    <1>     ;            (8 bit color, in BGR order, bit 0&1 is 0)
3263                    <1>     ;           ((EAX will return with color value))
3264                    <1>     ;           CL = DAC color register (index)
3265                    <1>     ;        BL = 7 : Set/Write single DAC color register
3266                    <1>     ;            (8 bit color, bit 0&1 are 0)
3267                    <1>     ;           ECX byte 0 - DAC color register
3268                    <1>     ;           ECX byte 1 - Blue (8 bit)
3269                    <1>     ;           ECX byte 2 - Green (8 bit)
3270                    <1>     ;           ECX byte 3 - Red (8 bit)
3271                    <1>     ;
3272                    <1>     ;       BL > 7 : invalid (not implemented)
3273                    <1>     ;
3274                    <1>     ;   if BL bit 2 is 1, 6 bit colors converted to 8 bit colors
3275                    <1>     ;     (low two bits of color bytes will be 0)
3276                    <1>     ;    ((color byte 00111111b will be converted to 11111100b))
3277                    <1>     ;     and RGB byte order will be
3278                    <1>     ;           byte 0 - Blue (low 2 bits are 0)
3279                    <1>     ;           byte 1 - Green (low 2 bits are 0)
3280                    <1>     ;           byte 2 - Red (low 2 bits are 0)
3281                    <1>     ;           byte 3 - pad (or zero byte)
3282                    <1>     ;     and 256 colors buffer size must be 256*4 = 1024 bytes
3283                    <1>     ;   if BL bit 2 is 0, 6 bit colors will be used directly
3284                    <1>     ;       (high two bits of 8 bit color bytes will be 0)
3285                    <1>     ;    ((dac color 111111b will be converted to 00111111b))
3286                    <1>     ;           byte 0 - Red (high 2 bits are 0)
3287                    <1>     ;           byte 1 - Green (high 2 bits are 0)
3288                    <1>     ;           byte 2 - Blue (high 2 bits are 0)
3289                    <1>     ;     and 256 colors buffer size must be 256*3 = 768 bytes
3290                    <1>     ;
3291                    <1>     ;     ECX = User's buffer addr (256*3 = 768 bytes) or
3292                    <1>     ;           Color
3293                    <1>     ;
```

```
3294                    <1>     ; Return:
3295                    <1>       ;    EAX = buffer size (for BL input = 0,1,4,5)
3296                    <1>     ;           or color value (for BL input = 2,3,6,7)
3297                    <1>     ;
3298                    <1>     ; 10/01/2021
3299                    <1>     ; SET/READ FONT DATA
3300                    <1>     ;
3301                    <1>     ;     BH = 12 = Set/Read Character Font Data
3302                    <1>     ;
3303                    <1>     ;        BL = 0 : Disable system font overwrite
3304                    <1>     ;        BL = 1 : Enable system font overwrite
3305                    <1>     ;        BL = 2 : Read system font 8x8
3306                    <1>     ;        BL = 3 : Read system font 8x14
3307                    <1>     ;        BL = 4 : Read system font 8x16
3308                    <1>     ;        BL = 5 : Read user defined font 8x8
3309                    <1>     ;        BL = 6 : Read user defined font 8x16
3310                    <1>     ;        BL = 7 : Write system font 8x8
3311                    <1>     ;        BL = 8 : Write system font 8x14
3312                    <1>     ;        BL = 9 : Write system font 8x16
3313                    <1>     ;        BL = 10 : Write user defined font 8x8
3314                    <1>     ;        BL = 11 : Write user defined font 8x16
3315                    <1>     ;
3316                    <1>     ;       BL > 11 : invalid (not implemented)
3317                    <1>     ;
3318                    <1>     ;     For BL = 1 to 11
3319                    <1>     ;        ECX = number of characters (<= 256)
3320                    <1>     ;        EDX = first character (ascii code in DL)
3321                    <1>     ;        ESI = user's buffer address
3322                    <1>     ;
3323                    <1>     ; Return:
3324                    <1>       ;    EAX = number of characters (ecx input)
3325                    <1>     ;     EAX = 0 -> error
3326                    <1>     ;     (EAX = 256 for BL = 0 and 1 if successful)
3327                    <1>     ;
3328                    <1>     ; Note: system font overwrite permission will be
3329                    <1>     ;       given if [multi_tasking] = 0
3330                    <1>     ;     and [u.uid] = 0 (BL = 1) ; 19/01/2021
3331                    <1>     ;       and if [ufont] bit 7 is 1 (BL = 7,8,9)
3332                    <1>     ;
3333                    <1>     ; 18/01/2021
3334                    <1>     ; SAVE/RESTORE STANDARD VGA VIDEO STATE
3335                    <1>     ;
3336                    <1>     ;     BH = 13 = Save/Restore std VGA video state
3337                    <1>     ;
3338                    <1>     ;        BL = 0 : Save VGA state (without DAC regs)
3339                    <1>     ;            Return: EAX = VideoStateID (>0)
3340                    <1>     ;                    EAX = 0 (failed!)
3341                    <1>     ;             (size: 110 bytes for TRDOS 386 v2.0.3)
3342                    <1>     ;        BL = 1 : Restore VGA state (without DAC regs)
3343                    <1>     ;            ECX = VideoStateID (to be verified)
3344                    <1>     ;              Return: EAX = Restore size (>0)
3345                    <1>     ;        BL = 2 : Save VGA state (complete)
3346                    <1>     ;            Return: EAX = VideoStateID (>0)
3347                    <1>     ;                    EAX = 0 (failed!)
3348                    <1>     ;             (size: 882 bytes for TRDOS 386 v2.0.3)
3349                    <1>     ;        BL = 3 : Restore VGA state (complete)
3350                    <1>     ;            ECX = VideoStateID (to be verified)
3351                    <1>     ;              Return: EAX = Restore size (>0)
3352                    <1>     ;
3353                    <1>     ;     * Above options are for saving
3354                    <1>     ;     *       video state to system memory
3355                    <1>     ;     *     (location: VBE3VIDEOSTATE, 2048 bytes)
3356                    <1>     ;
3357                    <1>     ;        BL = 4 : Save VGA state (without DAC regs)
3358                    <1>     ;            ECX = buffer address
3359                    <1>     ;            Return: EAX = transfer count
3360                    <1>     ;             (size: 110 bytes for TRDOS 386 v2.0.3)
3361                    <1>     ;            ECX = buffer address
3362                    <1>     ;        BL = 5 : Restore VGA state (without DAC regs)
3363                    <1>     ;            ECX = buffer address
3364                    <1>     ;              Return: EAX = transfer count
3365                    <1>     ;        BL = 6 : Save VGA state (complete)
3366                    <1>     ;            ECX = buffer address
3367                    <1>     ;            Return: EAX = transfer count
3368                    <1>     ;             (size: 882 bytes for TRDOS 386 v2.0.3)
3369                    <1>     ;         BL = 7 : Restore VGA state (complete)
3370                    <1>     ;            ECX = buffer address
3371                    <1>     ;              Return: EAX = transfer count
3372                    <1>     ;
3373                    <1>     ;     * Above options are for saving
3374                    <1>     ;     *       video state to user's buffer
3375                    <1>     ;     *     (buffer size: 110 bytes or 882 bytes)
3376                    <1>     ;
3377                    <1>     ;       BL > 7 : invalid (not implemented)
3378                    <1>     ;
3379                    <1>     ; 18/01/2021
3380                    <1>     ; SAVE/RESTORE SUPER VGA (VESA VBE 2/3) VIDEO STATE
3381                    <1>     ;
3382                    <1>     ;     BH = 14 = Save/Restore SVGA video state
3383                    <1>     ;
3384                    <1>     ;        BL = options
3385                    <1>     ;            bit 0 - Save (0) or Restore (1)
3386                    <1>     ;            bit 1 - controller hardware state
3387                    <1>     ;            bit 2 - BIOS data state
3388                    <1>     ;            bit 3 - DAC state
3389                    <1>     ;            bit 4 - (extended) Register state
3390                    <1>     ;            bit 5 - system (0) or user (1) memory
3391                    <1>     ;            bit 6 - verify without transfer
3392                    <1>     ;            bit 7 - not used (must be 0)
3393                    <1>     ;
3394                    <1>     ;          if bit 0 = 0 and bit 5 = 0
3395                    <1>     ;            Return:      EAX = VideoStateID (>0)
3396                    <1>     ;          if bit 0 = 1
3397                    <1>     ;            ECX = VideoStateID (bit 5 = 0)
3398                    <1>     ;            Return: EAX = restore (transfer) size
```

```
3399                                    <1>     ;           if bit 5 = 1
3400                                    <1>     ;             ECX = Buffer address
3401                                    <1>     ;             Return: EAX = transfer count (size)
3402                                    <1>     ;
3403                                    <1>     ;           ECX = Buffer address or VideoStateID
3404                                    <1>     ;
3405                                    <1>     ;         BL > 127 : invalid (not implemented)
3406                                    <1>     ;
3407                                    <1>     ;  Note: Required buffer size may be > 2048 bytes
3408                                    <1>     ;       (function fails when buff size > 2048 bytes)
3409                                    <1>     ;       proper option must be used
3410                                    <1>     ;
3411                                    <1>     ; 18/01/2021
3412                                    <1>     ; READ VESA EDID (EXTENDED DISPLAY IDENTIFICATION DATA)
3413                                    <1>     ;
3414                                    <1>     ;      BH = 15 = Read VESA EDID for connected monitor
3415                                    <1>     ;               (copy EDID to user)
3416                                    <1>     ;
3417                                    <1>     ;         BL = any
3418                                    <1>     ;
3419                                    <1>     ;      Input:
3420                                    <1>     ;          ECX = user's (EDID) buffer address
3421                                    <1>     ;              (buffer size: 128 bytes)
3422                                    <1>     ;      Output:
3423                                    <1>     ;          EAX = 128 (EDID size)
3424                                    <1>     ;          or EAX = 0 -> Error!
3425                                    <1>     ;             (EDID not ready or buffer addr error)
3426                                    <1>     ;
3427                                    <1>
3428                                    <1>     ; 16/05/2016
3429 0000E302 31C0                     <1>     xor    eax, eax
3430 0000E304 A3[64030300]             <1>     mov    [u.r0], eax
3431 0000E309 20FF                     <1>     and    bh, bh
3432 0000E30B 0F858B020000             <1>     jnz    sysvideo_13 ; 11/07/2016
3433                                    <1>
3434                                    <1>     ;; 21/11/2020 (TRDOS 386 v2.0.3)
3435                                    <1>     ;; tty/text mode transfers are only for video mode 3
3436                                    <1>
3437                                    <1>     ; 22/11/2020
3438                                    <1>     ;cmp   byte [CRT_MODE], 3 ; 80x25 text, 16 colors
3439                                    <1>     ;jne   sysret ; invalid (nothing to do), [u.r0] = 0
3440                                    <1>
3441                                    <1>     ; 23/11/2020
3442                                    <1>     ; bit 7,6,5,4 of BL are reserved and it must be 0
3443                                    <1>     ;             for current 'sysvideo' version
3444                                    <1>     ;test  bl, 0F0h
3445                                    <1>     ;;jnz  sysret ; invalid (undefined) !
3446                                    <1>     ;; 28/01/2021
3447                                    <1>     ;jnz   short sysvideo_1_2 ; invalid (undefined) !
3448                                    <1>     ; 28/01/2021
3449 0000E311 80FB07                   <1>     cmp    bl, 7
3450 0000E314 776E                     <1>     ja     short sysvideo_1_2 ; invalid (undefined) !
3451                                    <1>
3452                                    <1>     ; Video mode 0, 80*25 text mode, CGA 16 colors
3453                                    <1>     ; [CRT_MODE] = 3
3454                                    <1>     ;mov   bh, bl
3455                                    <1>     ;shr   bh, 2 ; 4..7 -> 1, 8..11 -> 2, 12..15 -> 3
3456                                    <1>     ;; 21/11/2020
3457                                    <1>     ;;and  bh, bh
3458                                    <1>       ;jnz sysvideo_4  ; Display page window transfer etc.
3459                                    <1>
3460                                    <1>     ; 28/01/2021
3461 0000E316 F6C304                   <1>     test   bl, 4 ; bit 2
3462 0000E319 0F85A2000000             <1>     jnz    sysvideo_4 ; Display page window transfer
3463                                    <1>
3464                                    <1>     ; Display page (complete) transfer
3465 0000E31F 80FA07                   <1>     cmp    dl, 7
3466                                    <1>     ;jnz   sysret ; invalid (nothing to do), [u.r0] = 0
3467 0000E322 760A                     <1>     jna    short sysvideo_0 ; 28/01/2021
3468 0000E324 FEC2                     <1>     inc    dl ; 0FFh -> 0 ("use current video page")
3469 0000E326 755C                     <1>     jnz    short sysvideo_1_2  ; invalid
3470                                    <1>     ; dl = 0 -> use current current page
3471 0000E328 8A15[468A0100]           <1>     mov    dl, [ACTIVE_PAGE]
3472                                    <1> sysvideo_0:
3473                                    <1>     ; 28/01/2021
3474 0000E32E 80FB03                   <1>     cmp    bl, 3
3475 0000E331 7206                     <1>     jb     short sysvideo_0_0
3476 0000E333 09FF                     <1>     or     edi, edi
3477 0000E335 744D                     <1>     jz     short sysvideo_1_2  ; invalid
3478 0000E337 89FE                     <1>     mov    esi, edi ; save swap/exchange buffer addr
3479                                    <1>     ; ecx = user buffer for new video page content
3480                                    <1>     ; esi = user (swap) buffer for saving current video page
3481                                    <1> sysvideo_0_0:
3482 0000E339 BF00800B00               <1>     mov    edi, 0B8000h
3483                                    <1>     ; dl = display page number, destination
3484 0000E33E 66B80010                 <1>     mov    ax, 4096 ; 21/11/2020
3485 0000E342 20D2                     <1>     and    dl, dl
3486 0000E344 7408                     <1>     jz     short sysvideo_1
3487                                    <1>     ; 07/02/2021
3488 0000E346 88D6                     <1>     mov    dh, dl
3489                                    <1> sysvideo_0_1:
3490                                    <1>     ; page length = 4096 bytes (but page content is 80*25*2 bytes)
3491 0000E348 01C7                     <1>     add    edi, eax ; 21//11/2020 ([CRT_LEN] = 1000h for mode 3)
3492 0000E34A FECE                     <1>     dec    dh
3493 0000E34C 75FA                     <1>     jnz    short sysvideo_0_1
3494                                    <1> sysvideo_1:
3495 0000E34E 80E303                   <1>     and    bl, 3
3496 0000E351 7536                     <1>     jnz    short sysvideo_2 ; user to system display page transfer
3497                                    <1>     ; system to system video page (content) transfer
3498                                    <1>     ; cl = display page number, source
3499 0000E353 80F907                   <1>     cmp    cl, 7
3500                                    <1>     ;ja    sysret ; invalid (nothing to do), [u.r0] = 0
3501 0000E356 760A                     <1>     jna    short sysvideo_1_0
3502 0000E358 FEC1                     <1>     inc    cl ; 0FFh -> 0 ("use current video page")
3503 0000E35A 7528                     <1>     jnz    short sysvideo_1_2  ; invalid
```

```
3504 0000E35C 8A0D[468A0100]      <1>       mov   cl, [ACTIVE_PAGE]
3505                              <1> sysvideo_1_0:
3506                              <1>       ; 28/01/2021
3507 0000E362 38D1                <1>       cmp   cl, dl
3508 0000E364 741E                <1>       je    short sysvideo_1_2  ; same video page !
3509                              <1>
3510                              <1>       ; system to system video/display page transfer (mode 0)
3511                              <1>       ; 21/11/2020
3512                              <1>       ;mov  esi, 0B8000h
3513                              <1>       ;movzx eax, cl
3514                              <1>       ;mov  edx, 4096 ; [CRT_LEN] = 1000h for video mode 3
3515                              <1>       ;mov  ecx, edx
3516                              <1>       ;mul  edx
3517                              <1>       ;add  esi, eax
3518                              <1>       ; 28/01/2021
3519                              <1>       ;movzx esi, cl
3520                              <1>       ;shl  si, 12 ; * 4096
3521                              <1>       ;add  esi, 0B8000h
3522                              <1>
3523                              <1>       ; 28/01/2021
3524 0000E366 A3[64030300]        <1>       mov   [u.r0], eax ; 4096
3525 0000E36B BE00800B00          <1>       mov   esi, 0B8000h
3526 0000E370 08C9                <1>       or    cl, cl
3527 0000E372 740A                <1>       jz    short sysvideo_1_1
3528                              <1>       ; 07/02/2021
3529 0000E374 88C8                <1>       mov   al, cl ; display/video page
3530 0000E376 30E4                <1>       xor   ah, ah
3531 0000E378 66C1E00C            <1>       shl   ax, 12 ; * 4096
3532 0000E37C 01C6                <1>       add   esi, eax
3533                              <1> sysvideo_1_1:
3534                              <1>       ; 21/11/2020
3535                              <1>       ;;mov  ecx, 4096
3536                              <1>       ;mov  ecx, eax ; 4096
3537                              <1>       ;;mov  [u.r0], ecx ; 4096 bytes
3538                              <1>       ; 28/01/2021
3539                              <1>       ;mov  [u.r0], cx
3540 0000E37E 31C9                <1>       xor   ecx, ecx
3541 0000E380 B504                <1>       mov   ch, 4 ; mov ecx, 1024
3542                              <1>       ;shr  cx, 2 ; / 4
3543 0000E382 F3A5                <1>       rep   movsd
3544                              <1> sysvideo_1_2:
3545 0000E384 E983F5FFFF          <1>       jmp   sysret
3546                              <1> sysvideo_2:
3547 0000E389 80FB02              <1>       cmp   bl, 2
3548                              <1>         ;ja  sysret; invalid (user to user), [u.r0] = 0
3549                              <1>       ; 28/01/2021
3550 0000E38C 7226                <1>       jb    short sysvideo_3 ; user to system
3551 0000E38E 7404                <1>       je    short sysvideo_2_0 ; system to user
3552                              <1>       ; bl = 3
3553 0000E390 89CA                <1>       mov   edx, ecx ; save user's buffer addr
3554 0000E392 89F1                <1>       mov   ecx, esi ; save swap address
3555                              <1> sysvideo_2_0:
3556                              <1>       ; bl = 2 (or bl = 3, stage 1)
3557                              <1>       ; system to user video/display page transfer (mode 0)
3558 0000E394 89FE                <1>       mov   esi, edi
3559 0000E396 89CF                <1>       mov   edi, ecx ; user buffer ; 28/01/2021
3560                              <1>       ; 21/11/2020
3561 0000E398 89C1                <1>       mov   ecx, eax ; 4096
3562 0000E39A E8A8370000          <1>       call  transfer_to_user_buffer ; fast transfer
3563                              <1>       ;jc   sysret ; [u.r0] = 0
3564 0000E39F 72E3                <1>       jc    short sysvideo_1_2 ; 28/01/2021
3565                              <1>       ; 28/01/2021
3566 0000E3A1 80FB03              <1>       cmp   bl, 3
3567 0000E3A4 7408                <1>       je    short sysvideo_2_2
3568                              <1> sysvideo_2_1:
3569                              <1>       ; 21/11/2020
3570 0000E3A6 890D[64030300]      <1>       mov   [u.r0], ecx
3571                              <1>       ;;mov  [u.r0], cx
3572                              <1>       ;jmp  sysret
3573 0000E3AC EBD6                <1>       jmp   short sysvideo_1_2
3574                              <1>
3575                              <1> sysvideo_2_2:
3576                              <1>       ; bl = 3 (exchange/swap) complete display page
3577                              <1>       ; esi = video page start address
3578                              <1>       ; edx = user's buffer address
3579                              <1>
3580                              <1>       ;mov  ecx, 4096
3581 0000E3AE 89F7                <1>       mov   edi, esi ; video page start address
3582 0000E3B0 89D6                <1>       mov   esi, edx ; user's (new page) buffer address
3583 0000E3B2 EB04                <1>       jmp   short sysvideo_2_3
3584                              <1> sysvideo_3:
3585                              <1>       ; bl = 1 (or bl = 3, stage 2)
3586                              <1>       ; user to system video/display page transfer (mode 0)
3587 0000E3B4 89CE                <1>       mov   esi, ecx ; user buffer
3588                              <1>       ; edi = video page address
3589                              <1>       ; 21/11/2020
3590 0000E3B6 89C1                <1>       mov   ecx, eax ; 4096
3591                              <1> sysvideo_2_3:
3592 0000E3B8 E8D4370000          <1>       call  transfer_from_user_buffer ; fast transfer
3593                              <1>       ;jc   sysret ; [u.r0] = 0
3594 0000E3BD 72C5                <1>       jc    short sysvideo_1_2 ; 28/01/2021
3595 0000E3BF EBE5                <1>       jmp   short sysvideo_2_1
3596                              <1>
3597                              <1>       ; 21/11/2020
3598                              <1>       ;mov  [u.r0], ecx
3599                              <1>       ;;mov  [u.r0], cx
3600                              <1>       ;;jmp  sysret
3601                              <1>       ;jmp  short sysvideo_1_2 ; 28/01/2021
3602                              <1>
3603                              <1> sysvideo_4:
3604                              <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
3605                              <1>
3606                              <1>       ; Display page window transfer etc.
3607 0000E3C1 80E303              <1>       and   bl, 3
3608 0000E3C4 0F85F7000000        <1>         jnz     sysvideo_9 ; user to system or system to user
```

```
3609                                    <1>       ; 21/11/2020
3610                                    <1>       ; system to system video/display page window transfer (mode 0)
3611 0000E3CA 81FE00800B00              <1>       cmp    esi, 0B8000h ; source buffer address (system)
3612 0000E3D0 0F8236F5FFFF              <1>       jb     sysret
3613 0000E3D6 81FE00000C00              <1>       cmp    esi, 0B8000h+(4096*8)
3614 0000E3DC 0F832AF5FFFF              <1>       jnb    sysret
3615 0000E3E2 81FF00800B00              <1>       cmp    edi, 0B8000h ; destination buffer address (system)
3616 0000E3E8 0F821EF5FFFF              <1>       jb     sysret
3617 0000E3EE 81FF00000C00              <1>         cmp     edi, 0B8000h+(4096*8)
3618 0000E3F4 0F8312F5FFFF              <1>       jnb    sysret
3619                                    <1>       ;
3620 0000E3FA 51                        <1>       push   ecx ; X1 and Y1 position - top left column, row
3621 0000E3FB 0FB7C1                    <1>       movzx  eax, cx ; top left column
3622                                    <1>       ; 21/11/2020
3623 0000E3FE C1E910                    <1>       shr    ecx, 16 ; top row
3624 0000E401 740E                      <1>       jz     short sysvideo_4_0 ; bypass following code
3625 0000E403 52                        <1>       push   edx ; X2 and Y2 position - bottom right column, row
3626 0000E404 50                        <1>       push   eax
3627 0000E405 66B8A000                  <1>       mov    ax, 80*2 ; 80 colums, 160 bytes per row
3628 0000E409 F7E1                      <1>       mul    ecx
3629                                    <1>              ; eax = offset for start row number
3630 0000E40B 01C6                      <1>       add    esi, eax
3631 0000E40D 01C7                      <1>       add    edi, eax
3632 0000E40F 58                        <1>       pop    eax
3633 0000E410 5A                        <1>       pop    edx
3634                                    <1> sysvideo_4_0:
3635 0000E411 66D1E0                    <1>       shl    ax, 1 ; * 2 ; convert start column number to offset
3636 0000E414 7404                      <1>       jz     short sysvideo_4_1
3637 0000E416 01C6                      <1>       add    esi, eax
3638 0000E418 01C7                      <1>       add    edi, eax
3639                                    <1>              ; esi = source page window start offset
3640                                    <1>              ; edi = destination page window start offset
3641                                    <1> sysvideo_4_1:
3642 0000E41A 59                        <1>       pop    ecx
3643                                    <1>       ;mov   eax, 0B8000h+(80*25*2*8)
3644                                    <1>       ; 21/11/2020
3645 0000E41B B800000C00                <1>       mov    eax, 0B8000h+(4096*8)
3646 0000E420 39C6                      <1>       cmp    esi, eax
3647 0000E422 0F83E4F4FFFF              <1>       jnb    sysret ; out of video page
3648 0000E428 39C6                      <1>       cmp    esi, eax
3649 0000E42A 0F83DCF4FFFF              <1>       jnb    sysret  ;out of video page
3650                                    <1>
3651 0000E430 56                        <1>       push   esi ; ****
3652 0000E431 57                        <1>       push   edi ; ***
3653 0000E432 52                        <1>       push   edx ; **
3654 0000E433 51                        <1>       push   ecx ; *
3655 0000E434 C1E910                    <1>       shr    ecx, 16 ; top row
3656 0000E437 C1EA10                    <1>       shr    edx, 16 ; bottom row
3657                                    <1>       ; 21/11/2020
3658                                    <1>       ;cmp   ecx, 24 ; max. 25 rows
3659 0000E43A 6683F918                  <1>       cmp    cx, 24
3660 0000E43E 7778                      <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
3661                                    <1>       ;cmp   edx, 24 ; max. 25 rows
3662 0000E440 6683FA18                  <1>       cmp    dx, 24
3663 0000E444 7772                      <1>       ja     short sysvideo_6 ; invalid, [u.r0] = 0
3664 0000E446 28CA                      <1>       sub    dl, cl ; end >= start
3665 0000E448 726E                      <1>       jc     short sysvideo_6 ; invalid, [u.r0] = 0
3666                                    <1>       ; 21/11/2020
3667 0000E44A 89D3                      <1>       mov    ebx, edx ; row count - 1
3668 0000E44C 7415                      <1>       jz     short sysvideo_4_2
3669 0000E44E 50                        <1>       push   eax ; *****
3670 0000E44F B8A0000000                <1>       mov    eax, 80*2 ; bytes per row
3671 0000E454 F7E3                      <1>       mul    ebx ; 21/11/2020
3672                                    <1>              ; eax = window end offset
3673                                    <1>              ; (for the last row, before adding column bytes)
3674 0000E456 01C6                      <1>       add    esi, eax
3675 0000E458 01C7                      <1>       add    edi, eax
3676 0000E45A 58                        <1>       pop    eax ; ***** ; mode 3 video memory end (0C000h)
3677 0000E45B 39C6                      <1>       cmp    esi, eax
3678                                    <1>       ;ja    short sysvideo_6 ; invalid, [u.r0] = 0
3679 0000E45D 7359                      <1>       jnb    short sysvideo_6 ; 21/11/2020
3680 0000E45F 39C7                      <1>       cmp    edi, eax
3681                                    <1>       ;ja    short sysvideo_6 ; invalid, [u.r0] = 0
3682 0000E461 7355                      <1>       jnb    short sysvideo_6 ; 21/11/2020
3683                                    <1> sysvideo_4_2:
3684 0000E463 59                        <1>       pop    ecx ; *
3685 0000E464 5A                        <1>       pop    edx ; **
3686 0000E465 81E1FFFF0000              <1>       and    ecx, 0FFFFh
3687 0000E46B 81E2FFFF0000              <1>       and    edx, 0FFFFh
3688                                    <1>       ; 21/11/2020
3689                                    <1>       ;cmp   ecx, 79 ; max. 80 columns
3690 0000E471 6683F94F                  <1>       cmp    cx, 79
3691 0000E475 7743                      <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
3692                                    <1>       ;cmp   edx, 79 ; max. 80 columns
3693 0000E477 6683FA4F                  <1>       cmp    dx, 79
3694 0000E47B 773D                      <1>       ja     short sysvideo_7 ; invalid, [u.r0] = 0
3695 0000E47D 28CA                      <1>       sub    dl, cl
3696 0000E47F 7639                      <1>       jna    short sysvideo_7 ; invalid, [u.r0] = 0
3697                                    <1>       ; 21/11/2020
3698 0000E481 740E                      <1>       jz     short sysvideo_4_3
3699                                    <1>       ; edx = column count (width) - 1
3700 0000E483 D0E2                      <1>       shl    dl, 1 ; * 2 ; byte offset (in end row)
3701 0000E485 01D6                      <1>       add    esi, edx
3702 0000E487 01D7                      <1>       add    edi, edx
3703                                    <1>              ; esi = source page window end offset
3704                                    <1>              ; edi = destination page window end offset
3705 0000E489 39C6                      <1>       cmp    esi, eax ; video memory end
3706                                    <1>       ;ja    short sysvideo_7
3707 0000E48B 732D                      <1>       jnb    short sysvideo_7 ; 21/11/2020
3708 0000E48D 39C7                      <1>       cmp    edi, eax ; video memory end
3709                                    <1>       ;ja    short sysvideo_7
3710 0000E48F 7329                      <1>       jnb    short sysvideo_7 ; 21/11/2020
3711                                    <1> sysvideo_4_3:
3712 0000E491 5F                        <1>       pop    edi ; ***
3713 0000E492 5E                        <1>       pop    esi ; ****
```

```
                                <1>       inc   bl ; row count - 1 -> row count
                                <1>       inc   dl ; column count
                                <1>       mov   bh, dl
                                <1>       shl   dl, 1 ; convert column count to byte offset
                                <1>       ; 21/11/2020
                                <1>       ; esi = source page window start offset
                                <1>       ; edi = destination page window start offset
0000E49B B8A0000000             <1>       mov   eax, 80*2 ; bytes per row
                                <1>       ; Note: 160 bytes per row (even if move count < 160)
                                <1> sysvideo_5:
0000E4A0 88F9                   <1>       mov   cl, bh ; move/transfer -word- count per row
0000E4A2 0115[64030300]         <1>       add   [u.r0], edx ; transfer count in bytes
0000E4A8 F366A5                 <1>       rep   movsw
0000E4AB 01C6                   <1>       add   esi, eax ; + 160 bytes to next row
0000E4AD 01C7                   <1>       add   edi, eax ; + 160 bytes to next row
0000E4AF FECB                   <1>       dec   bl ; remain count of rows
0000E4B1 75ED                   <1>       jnz   short sysvideo_5
0000E4B3 E954F4FFFF             <1>       jmp   sysret
                                <1>
                                <1> sysvideo_6:
0000E4B8 59                     <1>       pop   ecx ; *
0000E4B9 5A                     <1>       pop   edx ; **
                                <1> sysvideo_7:
0000E4BA 5F                     <1>       pop   edi ; ***
0000E4BB 5E                     <1>       pop   esi ; ****
                                <1> sysvideo_8:
0000E4BC E94BF4FFFF             <1>       jmp   sysret
                                <1>
                                <1> sysvideo_9:
                                <1>       ; user to system or system to user window transfer
                                <1>       ; 28/01/2021 (bl = 3 -> swap/exchange)
                                <1>       ;cmp   bl, 2
                                <1>       ;;ja   sysret ; user to user transfer is invalid
                                <1>       ;            ; [u.r0] = 0
                                <1>       ;ja    short sysvideo_8 ; 26/12/2020
                                <1>
                                <1>       ; 28/01/2021
0000E4C1 80FB02                 <1>       cmp   bl, 2
0000E4C4 7604                   <1>       jna   short sysvideo_9_8
                                <1>
                                <1>       ; swap/ exchange video memory and user mem windows
                                <1>       ; edi = swap address in user's memory space
0000E4C6 21FF                   <1>       and   edi, edi
0000E4C8 74F2                   <1>       jz    short sysvideo_8 ; invalid ; 28/01/2021
                                <1>
                                <1> sysvideo_9_8:
0000E4CA 56                     <1>       push  esi ; ****
0000E4CB 57                     <1>       push  edi ; ***
0000E4CC 52                     <1>       push  edx ; **
0000E4CD 51                     <1>       push  ecx ; *
                                <1>
0000E4CE C1E910                 <1>       shr   ecx, 16 ; top row
0000E4D1 C1EA10                 <1>       shr   edx, 16 ; bottom row
                                <1>
                                <1>       ; 21/11/2020
                                <1>       ;cmp   ecx, 24 ; max. 25 rows
0000E4D4 6683F918               <1>       cmp   cx, 24
0000E4D8 77DE                   <1>       ja    short sysvideo_6 ; invalid, [u.r0] = 0
                                <1>       ;cmp   edx, 24 ; max. 25 rows
0000E4DA 6683FA18               <1>       cmp   dx, 24
0000E4DE 77D8                   <1>       ja    short sysvideo_6 ; invalid, [u.r0] = 0
0000E4E0 28CA                   <1>       sub   dl, cl
0000E4E2 72D4                   <1>       jc    short sysvideo_6 ; invalid, [u.r0] = 0
                                <1>
                                <1>       ;mov   ch, cl ; top row
                                <1>       ;mov   cl, [ACTIVE_PAGE]
                                <1>
                                <1>       ;mov   edi, 80*25*2 ; 4000
                                <1>       ; 21/11/2020
                                <1>       ;mov   edi, 4096 ; [CRT_LEN = 4096 for video mode 3
                                <1>       ;shl   edi, cl  ; ! wrong for page 2 to page 7 !
                                <1>       ;;add  edi, 0B8000h - 80*25*2
                                <1>       ;add   edi, 0B8000h - 1000h ; - 4096
                                <1>
                                <1>       ; 21/11/2020
                                <1>       ;xor   eax, eax
                                <1>       ;mov   edi, 0B8000h
                                <1>       ;and   cl, cl ; is video page = 0 ?
                                <1>       ;jz    short sysvideo_9_1 ; yes
                                <1>       ; eax = 0
                                <1>
                                <1> ;sysvideo_9_0:
                                <1>       ;add   ax, 4096
                                <1>       ;dec   cl
                                <1>       ;jnz   short sysvideo_9_0
                                <1>       ;add   edi, eax
                                <1>       ;      ; edi = video page start address
                                <1>
                                <1>       ; 21/11/2020
0000E4E4 BF00800B00             <1>       mov   edi, 0B8000h
0000E4E9 803D[468A0100]00       <1>       cmp   byte [ACTIVE_PAGE], 0
0000E4F0 760C                   <1>       jna   short sysvideo_9_1
                                <1> stsvideo_9_0:
0000E4F2 B010                   <1>       mov   al, 16 ; 4096/256
0000E4F4 F625[468A0100]         <1>       mul   byte [ACTIVE_PAGE]
0000E4FA 86E0                   <1>       xchg  ah, al ; * 256
0000E4FC 01C7                   <1>       add   edi, eax
                                <1>             ; edi = video page start address
                                <1> sysvideo_9_1:
                                <1>       ; bl = transfer direction
                                <1>       ;     (1 = from user, 2 = to user)
                                <1>       ;     (3 = swap) ; 28/01/2021
0000E4FE 88D7                   <1>       mov   bh, dl ; row count - 1
                                <1>       ;mov   dl, ch ; top row
                                <1>       ; 21/11/2020
```

```
3819 0000E500 08C9              <1>       or    cl, cl ; top row number
3820 0000E502 7408              <1>       jz    short sysvideo_9_2
3821                            <1>
3822                            <1>       ;mov  eax, 80*2
3823 0000E504 66B8A000          <1>       mov   ax, 80*2 ; 160, bytes per row
3824 0000E508 F7E1              <1>       mul   ecx ; 22/11/2020
3825 0000E50A 01C7              <1>       add   edi, eax
3826                            <1>             ; edi = window start address for top row
3827                            <1> sysvideo_9_2:
3828 0000E50C 59                <1>       pop   ecx ; *
3829 0000E50D 5A                <1>       pop   edx ; **
3830 0000E50E 81E1FFFF0000      <1>       and   ecx, 0FFFFh
3831 0000E514 81E2FFFF0000      <1>       and   edx, 0FFFFh
3832                            <1>       ; 21/11/2020
3833                            <1>       ;cmp  ecx, 79 ; max. 80 columns
3834 0000E51A 6683F94F          <1>       cmp   cx, 79
3835 0000E51E 779A              <1>       ja    short sysvideo_7 ; invalid, [u.r0] = 0
3836                            <1>       ;cmp  edx, 79 ; max. 80 columns
3837 0000E520 6683FA4F          <1>       cmp   dx, 79
3838 0000E524 7794              <1>       ja    short sysvideo_7 ; invalid, [u.r0] = 0
3839                            <1>
3840 0000E526 28CA              <1>       sub   dl, cl
3841 0000E528 7290              <1>       jc    short sysvideo_7 ; invalid, [u.r0] = 0
3842                            <1>
3843 0000E52A 08C9              <1>       or    cl, cl ; left column
3844 0000E52C 7404              <1>       jz    short sysvideo_9_3 ; 0
3845                            <1>
3846                            <1>       ; 21/11/2020
3847 0000E52E D0E1              <1>       shl   cl, 1  ; column * 2
3848 0000E530 01CF              <1>       add   edi, ecx
3849                            <1>             ; edi = window start addr for top left column
3850                            <1> sysvideo_9_3:
3851 0000E532 88D1              <1>       mov   cl, dl
3852 0000E534 FEC1              <1>       inc   cl ; column count
3853 0000E536 D0E1              <1>       shl   cl, 1 ; column count * 2
3854                            <1>             ; ecx = transfer count per row
3855                            <1>
3856 0000E538 58                <1>       pop   eax ; *** (swap address)
3857 0000E539 5E                <1>       pop   esi ; ****
3858                            <1>
3859 0000E53A FEC7              <1>       inc   bh  ; row count
3860                            <1>
3861                            <1>       ;mov  edx, 80*2
3862 0000E53C B2A0              <1>       mov   dl, 80*2  ; bytes per row
3863                            <1>       ;
3864                            <1>       ;cmp  bl, 1 ; transfer direction
3865                            <1>       ;ja   short sysvideo_11 ; system to user transfer
3866                            <1>       ; 28/01/2021
3867 0000E53E F6C301            <1>       test  bl, 1
3868 0000E541 7439              <1>       jz    short sysvideo_11 ; system to user transfer
3869                            <1>
3870                            <1>       ; user to system video/display page window transfer (mode 0)
3871 0000E543 21C0              <1>       and   eax, eax ; swap address
3872 0000E545 741B              <1>       jz    short sysvideo_10 ; no window swap
3873                            <1> sysvideo_9_7: ; 28/01/2021
3874                            <1>       ; save previous window content in user's buffer (swap address)
3875 0000E547 56                <1>       push  esi ; user buffer
3876 0000E548 57                <1>       push  edi ; beginning address of the window
3877                            <1>       ; 21/11/2020
3878 0000E549 53                <1>       push  ebx ; save bh
3879 0000E54A 89FE              <1>       mov   esi, edi
3880 0000E54C 89C7              <1>       mov   edi, eax
3881                            <1> sysvideo_9_4:
3882 0000E54E E8F4350000        <1>       call  transfer_to_user_buffer ; fast transfer
3883 0000E553 7208              <1>       jc    short sysvideo_9_5
3884                            <1>       ; ecx = actual transfer count (must be same with input)
3885 0000E555 01D6              <1>       add   esi, edx ; next row address of (video page) window
3886 0000E557 01CF              <1>       add   edi, ecx ; next row address of user's window
3887                            <1>             ; Note: ecx may be less than row length of video page
3888                            <1>             ; user's window uses offset according to window width
3889 0000E559 FECF              <1>       dec   bh
3890 0000E55B 75F1              <1>       jnz   short sysvideo_9_4 ; repeat for next row
3891                            <1> sysvideo_9_5:
3892 0000E55D 5B                <1>       pop   ebx ; restore bh
3893 0000E55E 5F                <1>       pop   edi
3894 0000E55F 5E                <1>       pop   esi
3895                            <1>       ;jnc  short sysvideo_10
3896 0000E560 7215              <1>       jc    short sysvideo_9_6 ; 28/01/2021
3897                            <1> ;sysvideo_9_6:
3898                            <1> ;     jmp   sysret ; [u.r0] = 0
3899                            <1>
3900                            <1> sysvideo_10:
3901                            <1>       ; user to system video/display page window transfer (mode 0)
3902                            <1>       ; esi =      user buffer
3903 0000E562 E82A360000        <1>       call  transfer_from_user_buffer ; fast transfer
3904                            <1>       ;jc   sysret
3905 0000E567 720E              <1>       jc    short sysvideo_9_6 ; 28/01/2021
3906                            <1>       ; ecx = actual transfer count (must be same with input)
3907 0000E569 010D[64030300]    <1>       add   [u.r0], ecx ; actual transfer count
3908 0000E56F 01D7              <1>       add   edi, edx ; next row address of (video page) window
3909 0000E571 01CE              <1>       add   esi, ecx ; next row address of user's window
3910                            <1>             ; Note: ecx may be less than row length of video page
3911                            <1>             ; user's window uses offset according to window width
3912 0000E573 FECF              <1>       dec   bh
3913 0000E575 75EB              <1>       jnz   short sysvideo_10 ; repeat for next row
3914                            <1>       ;jmp  sysret
3915                            <1> sysvideo_9_6:
3916 0000E577 E990F3FFFF        <1>       jmp   sysret
3917                            <1>
3918                            <1> sysvideo_11:
3919                            <1>       ; system to user video/display page window transfer (mode 0)
3920 0000E57C 87FE              <1>       xchg  edi, esi
3921                            <1> sysvideo_12:
3922                            <1>       ; esi = beginning addr of the (screen, video page) window
3923                            <1>       ; edi =      user's buffer
```

```
3924 0000E57E E8C4350000          <1>       call   transfer_to_user_buffer ; fast transfer
3925 0000E583 0F8283F3FFFF        <1>       jc     sysret
3926                              <1>       ; ecx = actual transfer count (must be same with input)
3927 0000E589 010D[64030300]      <1>       add    [u.r0], ecx
3928 0000E58F 01D6                <1>       add    esi, edx ; next row (edx = 160)
3929 0000E591 01CF                <1>       add    edi, ecx ; next row of the user's window
3930                              <1>                       ; (ecx <= 160)
3931 0000E593 FECF                <1>       dec    bh
3932 0000E595 75E7                <1>       jnz    short sysvideo_12
3933                              <1> sysvideo_12_0:
3934 0000E597 E970F3FFFF          <1>       jmp    sysret
3935                              <1>
3936                              <1> sysvideo_13:
3937                              <1>       ; 28/12/2020
3938 0000E59C 80FF01              <1>       cmp    bh, 1
3939 0000E59F 7753                <1>         ja      short sysvideo_15 ; 23/11/2020
3940                              <1>
3941                              <1>       ; 25/02/2021
3942                              <1>       ; 12/02/2021
3943                              <1>       ; 29/01/2021, 31/01/2021
3944                              <1>       ; 23/11/2020 (TRDOS 386 v2.0.3)
3945                              <1>       ; (major modification, from mode 13h to all VGA modes,
3946                              <1>       ;  except super VGA modes and liner frame buffer method)
3947                              <1>
3948                              <1>       ; BH = 1 = VGA Graphics mode (0A0000h) data transfers
3949                              <1>
3950                              <1>       ; 29/01/2021
3951 0000E5A1 66B84001            <1>       mov    ax, 320      ; 320 pixels
3952 0000E5A5 F6C380              <1>       test   bl, 80h      ; bit 7 (screen width, 640 pixels)
3953 0000E5A8 7408                <1>       jz     short sysvideo_13_0
3954 0000E5AA 66D1E0              <1>       shl    ax, 1  ; 640 pixels
3955                              <1>       ;
3956 0000E5AD 80E37F              <1>       and    bl, 7Fh
3957 0000E5B0 7405                <1>       jz     short sysvideo_14
3958                              <1> sysvideo_13_0:
3959                              <1>       ; 29/01/2021
3960 0000E5B2 80FB41              <1>       cmp    bl, 41h
3961 0000E5B5 77E0                <1>       ja     short sysvideo_12_0 ; invalid (unknown) sub function
3962                              <1> sysvideo_14:
3963 0000E5B7 66A3[86120300]      <1>       mov    [v_width], ax; save screen width
3964 0000E5BD C705[8A120300]0000- <1>       mov    dword [v_mem], 0A0000h  ; save video memory address
3964 0000E5C5 0A00                <1>
3965 0000E5C7 C705[8E120300]0000- <1>       mov    dword [v_siz], 65536      ; save video memory size
3965 0000E5CF 0100                <1>
3966 0000E5D1 C705[96120300]0000- <1>       mov    dword [v_end], 0B0000h    ; save end of video page
3966 0000E5D9 0B00                <1>
3967 0000E5DB B708                <1>       mov    bh, 8
3968 0000E5DD 883D[89120300]      <1>       mov    [v_bpp], bh ; 8     ; bits per pixel (256 colors)
3969 0000E5E3 881D[88120300]      <1>       mov    [v_ops], bl  ; VGA data transfer options
3970                              <1>       ;mov   [maskbuff], edi ; 25/02/2021
3971 0000E5E9 893D[9A120300]      <1>       mov    [maskcolor], edi ; 25/02/2021
3972                              <1>                     ; save mask color or bitmask buffer address
3973 0000E5EF E9BD000000          <1>       jmp    sysvideo_15_7
3974                              <1>
3975                              <1> sysvideo_15:
3976                              <1>       ; 28/12/2020
3977 0000E5F4 80FF02              <1>       cmp    bh, 2
3978 0000E5F7 0F87CF1E0000        <1>         ja      sysvideo_16
3979                              <1>
3980                              <1>       ; 25/02/2021
3981                              <1>       ; 12/02/2021
3982                              <1>       ; 30/01/2021, 31/01/2021
3983                              <1>       ; 01/01/2021, 29/01/2021
3984                              <1>       ; 26/12/2020, 27/12/2020
3985                              <1>       ; 25/12/2020 (TRDOS 386 v2.0.3)
3986                              <1>       ;
3987                              <1>       ; BH = 2 = SVGA (VESA VBE) Graphics mode (LFB) data transfers
3988                              <1>
3989                              <1>       ; 25/12/2020
3990                              <1>       ; resolution table entry will be saved into EBP register
3991                              <1>
3992 0000E5FD 803D[5C090000]02    <1>       cmp    byte [vbe3], 2 ; VESA VBE 3 video bios
3993                              <1>                       ; or BOCHS/QEMU/VIRTUALBOX emu video bios
3994 0000E604 724E                <1>       jb     short sysvideo_15_4 ; no, nothing to do !
3995 0000E606 770B                <1>       ja     short sysvideo_15_0 ; yes
3996                              <1>
3997                              <1>       ; Only Bochs/Plex86 (emu) vbe2 video bios is usable in pmid
3998                              <1>       ; (if [vbe3] = 2)
3999 0000E608 A0[5D090000]        <1>       mov    al, [vbe2bios] ; Bochs vbios sign is from C0h to C5h
4000 0000E60D 24F0                <1>       and    al, 0F0h
4001 0000E60F 3CC0                <1>       cmp    al, 0C0h
4002 0000E611 7541                <1>       jne    short sysvideo_15_4 ; unknown (vbe2) video bios
4003                              <1> sysvideo_15_0:
4004                              <1>       ; 29/01/2021
4005 0000E613 80FB41              <1>       cmp    bl, 41h
4006 0000E616 773C                <1>       ja     short sysvideo_15_4 ; invalid (unknown) sub function
4007                              <1>       ; 29/01/2021
4008 0000E618 881D[88120300]      <1>       mov    [v_ops], bl  ; SVGA data transfer options
4009                              <1>
4010 0000E61E 89D8                <1>       mov    eax, ebx ; hw of ebx is vesa vbe video mode
4011 0000E620 C1E810              <1>       shr    eax, 16 ; ax = vesa vbe video mode
4012 0000E623 7513                <1>       jnz    short sysvideo_15_2
4013                              <1>       ; ax = 0
4014                              <1>
4015                              <1>       ; check & use current video mode
4016 0000E625 803D[DA6F0000]FF    <1>       cmp    byte [CRT_MODE], 0FFh ; extended (SVGA) mode ?
4017 0000E62C 7526                <1>       jne    short sysvideo_15_4 ; no
4018                              <1> sysvideo_15_1:
4019                              <1>       ; use current vbe (svga) video mode
4020 0000E62E 66A1[1E120300]      <1>       mov    ax, [video_mode] ; extended (SVGA, VESA VBE) mode
4021 0000E634 6625FF01            <1>       and    ax, 1FFh ; vesa vbe video mode: 1XXh
4022                              <1> sysvideo_15_2:
4023                              <1>       ; 29/01/2021
4024                              <1>       ;mov   [maskbuff], edi ; 25/02/2021
4025 0000E638 893D[9A120300]      <1>       mov    [maskcolor], edi ; 25/02/2021
```

```
4026                                    <1>                     ; save mask color or bitmask buffer address
4027 0000E63E BD[66730000]              <1>     mov     ebp, b_vbe_modes ; vbe mode table (in 'vidata.s')
4028                                    <1> sysvideo_15_3:
4029 0000E643 663B4500                  <1>     cmp     ax, [ebp]
4030 0000E647 7410                      <1>     je      short sysvideo_15_5
4031 0000E649 83C508                    <1>     add     ebp, 8 ; vbe mode table entry size
4032 0000E64C 81FD[26740000]            <1>     cmp     ebp, end_of_b_vbe_modes
4033 0000E652 72EF                      <1>     jb      short sysvideo_15_3
4034                                    <1> sysvideo_15_4:
4035                                    <1>     ; desired video mode is not a valid (implemented)
4036                                    <1>     ;       extended (VESA VBE, SVGA) video mode
4037                                    <1>     ;
4038                                    <1>     ; nothing to do !
4039                                    <1>
4040                                    <1>     ; [u.r0] = 0  ; return value of EAX
4041 0000E654 E9B3F2FFFF                <1>     jmp     sysret
4042                                    <1>
4043                                    <1> sysvideo_15_5:
4044                                    <1>     ; get LFB address
4045 0000E659 A1[2C120300]              <1>     mov     eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
4046 0000E65E 09C0                      <1>     or      eax, eax
4047 0000E660 7509                      <1>     jnz     short sysvideo_15_6
4048 0000E662 66A1[EB0E0000]            <1>     mov     ax, [def_LFB_addr] ; default LFB addr
4049                                    <1>                                ; (for vbe mode 118h)
4050 0000E668 C1E010                    <1>     shl     eax, 16
4051                                    <1>     ; 27/12/2020
4052                                    <1>     ;jz     short sysvideo_15_4
4053                                    <1> sysvideo_15_6:
4054                                    <1>     ; 29/01/2021
4055 0000E66B A3[8A120300]              <1>     mov     [v_mem], eax ; save video memory address
4056                                    <1>
4057                                    <1>     ; 27/12/2020
4058                                    <1>     ; 26/12/2020
4059 0000E670 8B4502                    <1>     mov     eax, [ebp+2] ; width, height
4060                                    <1>     ; 29/01/2021
4061 0000E673 66A3[86120300]            <1>     mov     [v_width], ax ; save screen width
4062                                    <1>     ; 28/12/2020
4063 0000E679 8A7D06                    <1>     mov     bh, [ebp+6] ; bpp
4064                                    <1>     ; 28/02/2021
4065                                    <1>     ; check default truecolor bpp value and use
4066                                    <1>     ; 32bpp instead of 24bpp if the default value
4067                                    <1>     ; has been set to 32bpp.
4068 0000E67C 80FF18                    <1>     cmp     bh, 24
4069 0000E67F 750B                      <1>     jne     short sysvideo_15_16
4070 0000E681 803D[DB860100]20          <1>     cmp     byte [truecolor], 32
4071                                    <1>             ; Default truecolor bpp value,
4072                                    <1>             ; it is 32 for VBE3 video bios
4073                                    <1>             ; (it can be set to 32 or 24)
4074 0000E688 7502                      <1>     jne     short sysvideo_15_16 ; not VBE3 !
4075                                    <1>                          ; or it is set to 24
4076 0000E68A B720                      <1>     mov     bh, 32
4077                                    <1>     ; 28/02/2021
4078                                    <1> sysvideo_15_16:
4079                                    <1>     ; 29/01/2021
4080 0000E68C 883D[89120300]            <1>     mov     [v_bpp], bh ; bits per pixel
4081                                    <1>
4082 0000E692 52                        <1>     push    edx ; *
4083 0000E693 0FB7D0                    <1>     movzx   edx, ax  ; width
4084 0000E696 C1E810                    <1>     shr     eax, 16  ; height
4085 0000E699 F7E2                      <1>     mul     edx
4086                                    <1>     ; eax = linear frame buffer size (pixels)
4087                                    <1>     ; 29/01/2021
4088 0000E69B A3[8E120300]              <1>     mov     [v_siz], eax ; save video page size
4089 0000E6A0 E8FD000000                <1>     call    pixels_to_byte_count
4090 0000E6A5 0305[8A120300]            <1>     add     eax, [v_mem]
4091 0000E6AB A3[96120300]              <1>     mov     [v_end], eax ; save end of video page
4092 0000E6B0 5A                        <1>     pop     edx ; *
4093                                    <1>
4094                                    <1>     ; bh = bits per pixel
4095                                    <1>     ; (bh will not be used after here, 29/01/2021)
4096                                    <1>
4097                                    <1>     ; bl = pixel operations & options
4098                                    <1>     ; ecx, edx, esi, edi input parameters
4099                                    <1>     ; [maskcolor] = edi input ; 25/02/2021
4100                                    <1>
4101                                    <1> sysvideo_15_7:
4102                                    <1>     ; 29/01/2021
4103                                    <1>     ;test  byte [v_ops], 40h  ; system to user ?
4104 0000E6B1 F6C340                    <1>     test    bl, 40h
4105 0000E6B4 7517                      <1>     jnz     short sysvideo_15_9
4106                                    <1>
4107 0000E6B6 31C0                      <1>     xor     eax, eax
4108 0000E6B8 88D8                      <1>     mov     al, bl
4109 0000E6BA BB[E0E70000]              <1>     mov     ebx, pixel_ops
4110 0000E6BF 240F                      <1>     and     al, 0Fh ; isolate 16 pixel operations
4111 0000E6C1 C0E002                    <1>     shl     al, 2 ; * 4 for dword table pointers
4112 0000E6C4 01C3                      <1>     add     ebx, eax
4113                                    <1>
4114                                    <1>     ; ebx = subroutine address
4115                                    <1>
4116                                    <1>     ; ecx, edx, esi, edi input parameters
4117                                    <1>     ; [maskbuff] = edi input
4118                                    <1>     ; [maskcolor] = edi input ; 25/02/2021
4119                                    <1>
4120 0000E6C6 FF13                      <1>     call    [ebx]
4121                                    <1> sysvideo_15_8:
4122 0000E6C8 E93FF2FFFF                <1>     jmp     sysret
4123                                    <1>
4124                                    <1> sysvideo_15_9:
4125                                    <1>     ; system to user display page or window copy
4126                                    <1>     ;test  byte [v_ops], 1 ; window copy ?
4127 0000E6CD F6C301                    <1>     test    bl, 1
4128 0000E6D0 7521                      <1>     jnz     short sysvideo_15_10
4129                                    <1>
4130                                    <1>     ; display page (full screen copy)
```

```
4131 0000E6D2 8B35[8A120300]      <1>       mov    esi, [v_mem] ; LFB start address
4132 0000E6D8 A1[8E120300]        <1>       mov    eax, [v_siz]
4133 0000E6DD E8C0000000          <1>       call   pixels_to_byte_count
4134 0000E6E2 89C1                <1>       mov    ecx, eax ; transfer count in bytes
4135                              <1>       ;edi = user's buffer address
4136 0000E6E4 E85E340000          <1>       call   transfer_to_user_buffer
4137 0000E6E9 72DD                <1>       jc     short sysvideo_15_8
4138 0000E6EB 890D[64030300]      <1>       mov    [u.r0], ecx
4139 0000E6F1 EBD5                <1>       jmp    short sysvideo_15_8
4140                              <1>
4141                              <1> sysvideo_15_10:
4142 0000E6F3 E820000000          <1>       call   sysvideo_15_12 ; window preparations
4143 0000E6F8 72CE                <1>       jc     short sysvideo_15_8
4144                              <1>
4145 0000E6FA 8B35[92120300]      <1>       mov    esi, [v_str]
4146                              <1> sysvideo_15_11:
4147                              <1>       ; esi = window's current row address (video mem)
4148                              <1>       ; edi = current row (virtual) addr in user's buff
4149                              <1>       ; ecx = transfer count per row
4150 0000E700 E842340000          <1>       call   transfer_to_user_buffer
4151 0000E705 72C1                <1>       jc     short sysvideo_15_8
4152 0000E707 010D[64030300]      <1>       add    [u.r0], ecx
4153 0000E70D 4B                  <1>       dec    ebx
4154 0000E70E 74B8                <1>       jz     short sysvideo_15_8 ; ok.
4155                              <1>       ; next row
4156 0000E710 01CF                <1>       add    edi, ecx ; next row in user's buffer
4157 0000E712 01D6                <1>       add    esi, edx ; next row of window (system)
4158 0000E714 EBEA                <1>       jmp    short sysvideo_15_11
4159                              <1>
4160                              <1> sysvideo_15_14:
4161 0000E716 F9                  <1>       stc     ; error !
4162                              <1> sysvideo_15_15:
4163 0000E717 C3                  <1>       retn
4164                              <1>
4165                              <1> sysvideo_15_12:
4166                              <1>       ; 30/01/2021
4167                              <1>       ; 29/01/2021
4168                              <1>       ; Window address preparations for window copy
4169 0000E718 6621D2              <1>       and    dx, dx
4170 0000E71B 74F9                <1>       jz     short sysvideo_15_14 ; invalid (zero columns)
4171                              <1>       ;test  edx, 0FFFF0000h
4172                              <1>       ;jz    short sysvideo_15_14 ; invalid (zero rows)
4173 0000E71D 81FA00000100        <1>       cmp    edx, 65536
4174 0000E723 72F2                <1>       jb     short sysvideo_15_15 ; invalid (zero rows)
4175 0000E725 89C8                <1>       mov    eax, ecx ; start position (row, column)
4176 0000E727 E899000000          <1>       call   calc_pixel_offset
4177 0000E72C 3B05[8E120300]      <1>       cmp    eax, [v_siz]
4178 0000E732 73E2                <1>       jnb    short sysvideo_15_14 ; out of display page
4179                              <1>                           ; nothing to do
4180 0000E734 E869000000          <1>       call   pixels_to_byte_count
4181 0000E739 0305[8A120300]      <1>       add    eax, [v_mem]
4182 0000E73F A3[92120300]        <1>       mov    [v_str], eax ; window start address
4183                              <1>                           ; (addr of top left corner)
4184                              <1>       ; check column limit
4185 0000E744 89C8                <1>       mov    eax, ecx
4186 0000E746 6601D0              <1>       add    ax, dx  ; add columns to start column
4187 0000E749 72CC                <1>       jc     short sysvideo_15_15 ; cf = 1
4188 0000E74B 663B05[86120300]    <1>       cmp    ax, [v_width]
4189 0000E752 77C2                <1>       ja     short sysvideo_15_14
4190                              <1>
4191 0000E754 89D0                <1>       mov    eax, edx ; size
4192 0000E756 2D00000100          <1>       sub    eax, 65536 ; row count -> 0 based row #
4193 0000E75B E865000000          <1>       call   calc_pixel_offset
4194 0000E760 3B05[8E120300]      <1>       cmp    eax, [v_siz] ; video (display) page size
4195 0000E766 77AE                <1>       ja     short sysvideo_15_14 ; out of display page
4196                              <1>                           ; nothing to do
4197 0000E768 E835000000          <1>       call   pixels_to_byte_count
4198 0000E76D 0305[92120300]      <1>       add    eax, [v_str] ; window start address
4199 0000E773 3B05[96120300]      <1>       cmp    eax, [v_end] ; window end address (+1)
4200                              <1>                      ; (addr of bottom right corner +1)
4201 0000E779 779B                <1>       ja     short sysvideo_15_14 ; out of display page
4202                              <1>                           ; nothing to do
4203 0000E77B 89D3                <1>       mov    ebx, edx
4204 0000E77D C1EB10              <1>       shr    ebx, 16
4205                              <1>       ; ebx = row count
4206 0000E780 81E2FFFF0000        <1>       and    edx, 0FFFFh
4207                              <1>       ; edx = transfer count per row (from user's buffer)
4208                              <1>       ;      (in pixels, window width)
4209 0000E786 89D0                <1>       mov    eax, edx
4210 0000E788 A3[9E120300]        <1>       mov    [pixcount], eax ; 27/02/2021
4211 0000E78D E810000000          <1>       call   pixels_to_byte_count
4212 0000E792 89C1                <1>       mov    ecx, eax
4213                              <1>       ; ecx = transfer count per row (from user's buffer)
4214                              <1>       ;      (in bytes, window width)
4215 0000E794 66A1[86120300]      <1>       mov    ax, [v_width]
4216 0000E79A E803000000          <1>       call   pixels_to_byte_count
4217 0000E79F 89C2                <1>       mov    edx, eax
4218                              <1>       ; edx = byte count per row
4219 0000E7A1 C3                  <1>       retn ; cf = 0
4220                              <1>
4221                              <1> pixels_to_byte_count:
4222                              <1>       ; 29/01/2021
4223                              <1>       ; INPUT:
4224                              <1>       ;   eax = pixel count
4225                              <1>       ; OUTPUT:
4226                              <1>       ;   eax = byte count
4227                              <1>       ;
4228 0000E7A2 803D[89120300]08    <1>       cmp    byte [v_bpp], 8
4229 0000E7A9 7619                <1>       jna    short pixtobc_3 ; 8 bit colors
4230 0000E7AB 803D[89120300]18    <1>       cmp    byte [v_bpp], 24
4231 0000E7B2 720A                <1>       jb     short pixtobc_1 ; 16 bit colors
4232 0000E7B4 770B                <1>       ja     short pixtobc_2 ; 32 bit colors
4233                              <1>       ; 24 bit pixels
4234                              <1>       ; eax = eax * 3
4235                              <1>       ;push  edx
```

```
4236                                    <1>       ;mov   edx, eax
4237                                    <1>       ;shl   eax, 1
4238                                    <1>       ;add   eax, edx
4239                                    <1>       ;pop   edx
4240 0000E7B6 50                       <1>       push   eax
4241 0000E7B7 D1E0                     <1>       shl    eax, 1
4242 0000E7B9 010424                   <1>       add    [esp], eax
4243 0000E7BC 58                       <1>       pop    eax
4244 0000E7BD C3                       <1>       retn
4245                                    <1> pixtobc_1:
4246                                    <1>       ; 32 bit pixels
4247                                    <1>       ; eax = eax * 2
4248 0000E7BE D1E0                     <1>       shl    eax, 1
4249 0000E7C0 C3                       <1>       retn
4250                                    <1> pixtobc_2:
4251                                    <1>       ; 16 bit pixels
4252                                    <1>       ; eax = eax * 4
4253 0000E7C1 C1E002                   <1>       shl    eax, 2
4254                                    <1> pixtobc_3:
4255 0000E7C4 C3                       <1>       retn
4256                                    <1>
4257                                    <1> calc_pixel_offset:
4258                                    <1>       ; 29/01/2021
4259                                    <1>       ; INPUT:
4260                                    <1>       ;   eax = pixel position (row, column)
4261                                    <1>       ; OUTPUT:
4262                                    <1>       ;   eax = pixel offset (linear address)
4263                                    <1>       ;
4264 0000E7C5 52                       <1>       push   edx
4265 0000E7C6 50                       <1>       push   eax
4266 0000E7C7 C1E810                   <1>       shr    eax, 16
4267 0000E7CA 7409                     <1>       jz     short cpixo_0
4268                                    <1>       ; eax = row
4269 0000E7CC 0FB715[86120300]         <1>       movzx  edx, word [v_width]
4270 0000E7D3 F7E2                     <1>       mul    edx
4271                                    <1> cpixo_0:
4272                                    <1>       ; eax = row * screen width
4273 0000E7D5 5A                       <1>       pop    edx
4274 0000E7D6 81E2FFFF0000             <1>       and    edx, 0FFFFh
4275                                    <1>       ; edx = column
4276 0000E7DC 01D0                     <1>       add    eax, edx
4277                                    <1>       ; eax = (row * screen width) + column
4278 0000E7DE 5A                       <1>       pop    edx
4279 0000E7DF C3                       <1>       retn
4280                                    <1>
4281                                    <1>       ; 02/02/2021
4282                                    <1>       ; 29/01/2021
4283                                    <1> pixel_ops:
4284 0000E7E0 [20E80000]               <1>       dd     pix_op_cpy ; copy pixels (user to system)
4285 0000E7E4 [6BED0000]               <1>       dd     pix_op_new ; change (new, fill) color
4286 0000E7E8 [88E80000]               <1>       dd     pix_op_add ; add color (up to 0FFh)
4287 0000E7EC [3AE90000]               <1>       dd     pix_op_sub ; sub color (down to 0)
4288 0000E7F0 [55EB0000]               <1>       dd     pix_op_orc ; or color
4289 0000E7F4 [07EC0000]               <1>       dd     pix_op_and ; and color
4290 0000E7F8 [B9EC0000]               <1>       dd     pix_op_xor ; xor color
4291 0000E7FC [2FEE0000]               <1>       dd     pix_op_not ; not color
4292 0000E800 [D9EE0000]               <1>       dd     pix_op_neg ; neg color
4293 0000E804 [83EF0000]               <1>       dd     pix_op_inc ; inc color
4294 0000E808 [2DF00000]               <1>       dd     pix_op_dec ; dec color
4295 0000E80C [ECE90000]               <1>       dd     pix_op_mix ; mix color
4296 0000E810 [B3EA0000]               <1>       dd     pix_op_rpl ; replace color
4297 0000E814 [D7F00000]               <1>       dd     pix_op_blk ; copy pixel block(s) (sys)
4298 0000E818 [86F10000]               <1>       dd     pix_op_lin ; write line(s)
4299 0000E81C [69F50000]               <1>       dd     pix_op_chr ; write character (font)
4300                                    <1>
4301                                    <1> pix_op_cpy:
4302                                    <1>       ; 21/02/2021
4303                                    <1>       ; 06/02/2021
4304                                    <1>       ; 30/01/2021
4305                                    <1>       ; COPY PIXELS
4306                                    <1>       ;
4307                                    <1>       ; INPUT:
4308                                    <1>       ;  If bit 4 of BL or [v_ops] = 1 -window copy-
4309                                    <1>       ;  ECX = start position (row, column)
4310                                    <1>       ;         (HW = row, CX = column)
4311                                    <1>       ;  EDX = size (rows, colums)
4312                                    <1>       ;         (HW = rows, DX = columns)
4313                                    <1>       ;       (0 -> invalid
4314                                    <1>       ;         (1 -> horizontal or vertical line)
4315                                    <1>       ;  If bit 4 of BL or [v_ops] = 0 -full screen-
4316                                    <1>       ;     ECX and EDX will not be used
4317                                    <1>       ;  ESI = user's buffer address
4318                                    <1>       ;  [maskcolor] = mask color (to be excluded)
4319                                    <1>       ;
4320                                    <1>       ; OUTPUT:
4321                                    <1>       ;      [u.r0] will be > 0 if succesful
4322                                    <1>
4323 0000E820 F605[88120300]10         <1>       test   byte [v_ops], 10h ; display page or window ?
4324 0000E827 752E                     <1>       jnz    short pix_op_cpy_w ; window
4325                                    <1>
4326 0000E829 8B3D[8A120300]           <1>       mov    edi, [v_mem] ; 21/02/2021
4327                                    <1>
4328                                    <1>       ; Copy user's buffer content do display page
4329                                    <1>       ; (full screen copy)
4330 0000E82F A1[8E120300]             <1>       mov    eax, [v_siz] ; video page size
4331 0000E834 E869FFFFFF               <1>       call   pixels_to_byte_count
4332 0000E839 89C1                     <1>       mov    ecx, eax ; transfer count
4333                                    <1>       ; esi = user's buffer address (virtual)
4334 0000E83B F605[88120300]20         <1>       test   byte [v_ops], 20h ; masked copy ?
4335 0000E842 7405                     <1>       jz     short pix_op_cpy_0 ; no
4336 0000E844 E9720F0000               <1>       jmp    m_pix_op_cpy ; copy pixels except mask color
4337                                    <1> pix_op_cpy_0:
4338                                    <1>       ; esi = user buffer for full screen copy
4339                                    <1>       ; edi = start of video memory
4340                                    <1>       ;      (start of display page)
```

```
                                <1>       ; ecx = byte count (display page size in bytes)
0000E849 E843330000             <1>       call   transfer_from_user_buffer
0000E84E 7206                   <1>       jc     short pix_op_cpy_1
0000E850 890D[64030300]         <1>       mov    [u.r0], ecx
                                <1> pix_op_cpy_1:
0000E856 C3                     <1>       retn   ; 06/02/2021
                                <1>
                                <1> pix_op_cpy_w:
0000E857 E8BCFEFFFF             <1>       call   sysvideo_15_12 ; window preparations
0000E85C 72F8                   <1>       jc     short pix_op_cpy_1
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
0000E85E 8B3D[92120300]         <1>       mov    edi, [v_str]
0000E864 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked copy ?
0000E86B 7405                   <1>       jz     short pix_op_cpy_w_0 ; no
0000E86D E90C100000             <1>       jmp    m_pix_op_cpy_w ; window copy except mask color
                                <1> pix_op_cpy_w_0:
                                <1>       ; esi = current row (virtual) addr in user's buff
                                <1>       ; edi = window's current row address (video mem)
                                <1>       ; ecx = transfer count per row
0000E872 E81A330000             <1>       call   transfer_from_user_buffer
0000E877 72DD                   <1>       jc     short pix_op_cpy_1
0000E879 010D[64030300]         <1>       add    [u.r0], ecx
0000E87F 4B                     <1>       dec    ebx
0000E880 74D4                   <1>       jz     short pix_op_cpy_1 ; ok.
                                <1>       ; next row
0000E882 01CE                   <1>       add    esi, ecx ; next row in user's buffer
0000E884 01D7                   <1>       add    edi, edx ; next row of window (system)
0000E886 EBEA                   <1>       jmp    short pix_op_cpy_w_0
                                <1>
                                <1> pix_op_add:
                                <1>       ; 31/01/2021
                                <1>       ; 30/01/2021
                                <1>       ; ADD 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>
0000E888 F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000E88F 7555                   <1>       jnz    short pix_op_add_w ; window
                                <1>
0000E891 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000E897 89FE                   <1>       mov    esi, edi
                                <1>       ; ecx = color (CL, CX, ECX)
0000E899 89C8                   <1>       mov    eax, ecx
0000E89B 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000E8A1 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color adding ?
0000E8A8 7405                   <1>       jz     short pix_op_add_0 ; no
0000E8AA E9CE100000             <1>       jmp    m_pix_op_add ; add color except mask color
                                <1> pix_op_add_0:
0000E8AF 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E8B6 7707                   <1>       ja     short pix_op_add_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E8B8 E84D0A0000             <1>       call   pix_op_add_8
0000E8BD EB1E                   <1>       jmp    short pix_op_add_4
                                <1>
                                <1> pix_op_add_1:
0000E8BF 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E8C6 7710                   <1>       ja     short pix_op_add_3 ; 32bpp
0000E8C8 7207                   <1>       jb     short pix_op_add_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000E8CA E85B0A0000             <1>       call   pix_op_add_24
0000E8CF EB0C                   <1>       jmp    short pix_op_add_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_add_2:
0000E8D1 E8420A0000             <1>       call   pix_op_add_16
0000E8D6 EB05                   <1>       jmp    short pix_op_add_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_add_3:
0000E8D8 E86D0A0000             <1>       call   pix_op_add_32
                                <1> pix_op_add_4:
0000E8DD 29F7                   <1>       sub    edi, esi
0000E8DF 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_add_5:
0000E8E5 C3                     <1>       retn
                                <1>
                                <1> pix_op_add_w:
                                <1>       ; 31/01/2021
0000E8E6 51                     <1>       push   ecx ; * ; color
0000E8E7 89D1                   <1>       mov    ecx, edx ; win start pos
0000E8E9 89F2                   <1>       mov    edx, esi ; size (rows, cols)
0000E8EB E828FEFFFF             <1>       call   sysvideo_15_12 ; window preparations
0000E8F0 58                     <1>       pop    eax ; * ; color
0000E8F1 72F2                   <1>       jc     short pix_op_add_5
                                <1>
0000E8F3 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color adding ?
0000E8FA 7405                   <1>       jz     short pix_op_add_w_0 ; no
0000E8FC E92A110000             <1>       jmp    m_pix_op_add_w
                                <1>                    ; window add color except mask color
```

```
                                <1> pix_op_add_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>
0000E901 8B3D[92120300]         <1>       mov    edi, [v_str]
0000E907 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E90E 7707                   <1>       ja     short pix_op_add_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E910 BD[0AF30000]           <1>       mov    ebp, pix_op_add_8
0000E915 EB1E                   <1>       jmp    short pix_op_add_w_4
                                <1>
                                <1> pix_op_add_w_1:
0000E917 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E91E 7710                   <1>       ja     short pix_op_add_w_3 ; 32bpp
0000E920 7207                   <1>       jb     short pix_op_add_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000E922 BD[2AF30000]           <1>       mov    ebp, pix_op_add_24
0000E927 EB0C                   <1>       jmp    short pix_op_add_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_add_w_2:
0000E929 BD[18F30000]           <1>       mov    ebp, pix_op_add_16
0000E92E EB05                   <1>       jmp    short pix_op_add_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_add_w_3:
0000E930 BD[4AF30000]           <1>       mov    ebp, pix_op_add_32
                                <1> pix_op_add_w_4:
0000E935 E95F010000             <1>       jmp    pix_op_add_w_x
                                <1>
                                <1> pix_op_sub:
                                <1>       ; 31/01/2021
                                <1>       ; SUB 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>
0000E93A F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000E941 7555                   <1>       jnz    short pix_op_sub_w ; window
                                <1>
0000E943 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000E949 89FE                   <1>       mov    esi, edi
                                <1>       ; ecx = color (CL, CX, ECX)
0000E94B 89C8                   <1>       mov    eax, ecx
0000E94D 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000E953 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color subtract ?
0000E95A 7405                   <1>       jz     short pix_op_sub_0 ; no
0000E95C E9FD100000             <1>       jmp    m_pix_op_sub ; sub color except mask color
                                <1> pix_op_sub_0:
0000E961 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000E968 7707                   <1>       ja     short pix_op_sub_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000E96A E8EA090000             <1>       call   pix_op_sub_8
0000E96F EB1E                   <1>       jmp    short pix_op_sub_4
                                <1>
                                <1> pix_op_sub_1:
0000E971 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000E978 7710                   <1>       ja     short pix_op_sub_3 ; 32bpp
0000E97A 7207                   <1>       jb     short pix_op_sub_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000E97C E8FB090000             <1>       call   pix_op_sub_24
0000E981 EB0C                   <1>       jmp    short pix_op_sub_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_sub_2:
0000E983 E8E1090000             <1>       call   pix_op_sub_16
0000E988 EB05                   <1>       jmp    short pix_op_sub_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_sub_3:
0000E98A E8070A0000             <1>       call   pix_op_sub_32
                                <1> pix_op_sub_4:
0000E98F 29F7                   <1>       sub    edi, esi
0000E991 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_sub_5:
0000E997 C3                     <1>       retn
                                <1>
                                <1> pix_op_sub_w:
                                <1>       ; 31/01/2021
0000E998 51                     <1>       push   ecx ; * ; color
0000E999 89D1                   <1>       mov    ecx, edx ; win start pos
0000E99B 89F2                   <1>       mov    edx, esi ; size (rows, cols)
0000E99D E876FDFFFF             <1>       call   sysvideo_15_12 ; window preparations
0000E9A2 58                     <1>       pop    eax ; * ; color
0000E9A3 72F2                   <1>       jc     short pix_op_sub_5
                                <1>
0000E9A5 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color subtract ?
0000E9AC 7405                   <1>       jz     short pix_op_sub_w_0 ; no
```

```
4551 0000E9AE E94E110000          <1>       jmp    m_pix_op_sub_w
4552                              <1>                      ; window sub color except mask color
4553                              <1> pix_op_sub_w_0:
4554                              <1>       ; ecx = bytes per row (to be applied)
4555                              <1>       ; edx = screen width in bytes
4556                              <1>       ; ebx = row count
4557                              <1>       ; eax = color
4558                              <1>
4559 0000E9B3 8B3D[92120300]      <1>       mov    edi, [v_str]
4560 0000E9B9 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
4561 0000E9C0 7707                <1>       ja     short pix_op_sub_w_1
4562                              <1>
4563                              <1>       ; 256 colors (8bpp)
4564 0000E9C2 BD[59F30000]        <1>       mov    ebp, pix_op_sub_8
4565 0000E9C7 EB1E                <1>       jmp    short pix_op_sub_w_4
4566                              <1>
4567                              <1> pix_op_sub_w_1:
4568 0000E9C9 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
4569 0000E9D0 7710                <1>       ja     short pix_op_sub_w_3 ; 32bpp
4570 0000E9D2 7207                <1>       jb     short pix_op_sub_w_2 ; 16bpp
4571                              <1>
4572                              <1>       ; 24 bit true colors
4573 0000E9D4 BD[7CF30000]        <1>       mov    ebp, pix_op_sub_24
4574 0000E9D9 EB0C                <1>       jmp    short pix_op_sub_w_4
4575                              <1>
4576                              <1>       ; 65536 colors (16bpp)
4577                              <1> pix_op_sub_w_2:
4578 0000E9DB BD[69F30000]        <1>       mov    ebp, pix_op_sub_16
4579 0000E9E0 EB05                <1>       jmp    short pix_op_sub_w_4
4580                              <1>
4581                              <1>       ; 32 bit true colors
4582                              <1> pix_op_sub_w_3:
4583 0000E9E2 BD[96F30000]        <1>       mov    ebp, pix_op_sub_32
4584                              <1> pix_op_sub_w_4:
4585 0000E9E7 E9AD000000          <1>       jmp    pix_op_sub_w_x
4586                              <1>
4587                              <1> pix_op_mix:
4588                              <1>       ; 31/01/2021
4589                              <1>       ; MIX COLOR
4590                              <1>       ;
4591                              <1>       ; INPUT:
4592                              <1>       ;   CL = color (8 bit, 256 colors)
4593                              <1>       ;  ECX = color (16 bit and true colors)
4594                              <1>       ;  EDX = start position (row, column)
4595                              <1>       ;         (HW = row, DX = column)
4596                              <1>       ;  ESI = size (rows, colums)
4597                              <1>       ;         (HW = rows, SI = columns)
4598                              <1>       ;
4599                              <1>       ;  [maskcolor] = mask color (to be excluded)
4600                              <1>       ;
4601                              <1>       ; OUTPUT:
4602                              <1>       ;      [u.r0] will be > 0 if succesful
4603                              <1>
4604 0000E9EC F605[88120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
4605 0000E9F3 7555                <1>       jnz    short pix_op_mix_w ; window
4606                              <1>
4607 0000E9F5 8B3D[8A120300]      <1>       mov    edi, [v_mem]
4608 0000E9FB 89FE                <1>       mov    esi, edi
4609                              <1>       ; ecx = color (CL, CX, ECX)
4610 0000E9FD 89C8                <1>       mov    eax, ecx
4611 0000E9FF 8B0D[8E120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
4612                              <1>
4613 0000EA05 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color mix ?
4614 0000EA0C 7405                <1>       jz     short pix_op_mix_0 ; no
4615 0000EA0E E921110000          <1>       jmp    m_pix_op_mix ; mix colors except mask color
4616                              <1> pix_op_mix_0:
4617 0000EA13 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
4618 0000EA1A 7707                <1>       ja     short pix_op_mix_1
4619                              <1>
4620                              <1>       ; 256 colors (8bpp)
4621 0000EA1C E8F4090000          <1>       call   pix_op_mix_8
4622 0000EA21 EB1E                <1>       jmp    short pix_op_mix_4
4623                              <1>
4624                              <1> pix_op_mix_1:
4625 0000EA23 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
4626 0000EA2A 7710                <1>       ja     short pix_op_mix_3 ; 32bpp
4627 0000EA2C 7207                <1>       jb     short pix_op_mix_2 ; 16bpp
4628                              <1>
4629                              <1>       ; 24 bit true colors
4630 0000EA2E E8FD090000          <1>       call   pix_op_mix_24
4631 0000EA33 EB0C                <1>       jmp    short pix_op_mix_4
4632                              <1>
4633                              <1>       ; 65536 colors (16bpp)
4634                              <1> pix_op_mix_2:
4635 0000EA35 E8E7090000          <1>       call   pix_op_mix_16
4636 0000EA3A EB05                <1>       jmp    short pix_op_mix_4
4637                              <1>
4638                              <1>       ; 32 bit true colors
4639                              <1> pix_op_mix_3:
4640 0000EA3C E80B0A0000          <1>       call   pix_op_mix_32
4641                              <1> pix_op_mix_4:
4642 0000EA41 29F7                <1>       sub    edi, esi
4643 0000EA43 893D[64030300]      <1>       mov    [u.r0], edi
4644                              <1> pix_op_mix_5:
4645 0000EA49 C3                  <1>       retn
4646                              <1>
4647                              <1> pix_op_mix_w:
4648                              <1>       ; 31/01/2021
4649 0000EA4A 51                  <1>       push   ecx ; * ; color
4650 0000EA4B 89D1                <1>       mov    ecx, edx ; win start pos
4651 0000EA4D 89F2                <1>       mov    edx, esi ; size (rows, cols)
4652 0000EA4F E8C4FCFFFF          <1>       call   sysvideo_15_12 ; window preparations
4653 0000EA54 58                  <1>       pop    eax ; * ; color
4654 0000EA55 72F2                <1>       jc     short pix_op_mix_5
4655                              <1>
```

```
4656 0000EA57 F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color mix ?
4657 0000EA5E 7405                <1>       jz     short pix_op_mix_w_0 ; no
4658 0000EA60 E96C110000          <1>       jmp    m_pix_op_mix_w
4659                              <1>                    ; window mix colors except mask color
4660                              <1> pix_op_mix_w_0:
4661                              <1>       ; ecx = bytes per row (to be applied)
4662                              <1>       ; edx = screen width in bytes
4663                              <1>       ; ebx = row count
4664                              <1>       ; eax = color
4665                              <1>
4666 0000EA65 8B3D[92120300]      <1>       mov    edi, [v_str]
4667 0000EA6B 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
4668 0000EA72 7707                <1>       ja     short pix_op_mix_w_1
4669                              <1>
4670                              <1>       ; 256 colors (8bpp)
4671 0000EA74 BD[15F40000]        <1>       mov    ebp, pix_op_mix_8
4672 0000EA79 EB1E                <1>       jmp    short pix_op_mix_w_x
4673                              <1>
4674                              <1> pix_op_mix_w_1:
4675 0000EA7B 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
4676 0000EA82 7710                <1>       ja     short pix_op_mix_w_3 ; 32bpp
4677 0000EA84 7207                <1>       jb     short pix_op_mix_w_2 ; 16bpp
4678                              <1>
4679                              <1>       ; 24 bit true colors
4680 0000EA86 BD[30F40000]        <1>       mov    ebp, pix_op_mix_24
4681 0000EA8B EB0C                <1>       jmp    short pix_op_mix_w_x
4682                              <1>
4683                              <1>       ; 65536 colors (16bpp)
4684                              <1> pix_op_mix_w_2:
4685 0000EA8D BD[21F40000]        <1>       mov    ebp, pix_op_mix_16
4686 0000EA92 EB05                <1>       jmp    short pix_op_mix_w_x
4687                              <1>
4688                              <1>       ; 32 bit true colors
4689                              <1> pix_op_mix_w_3:
4690 0000EA94 BD[4CF40000]        <1>       mov    ebp, pix_op_mix_32
4691                              <1>       ;jmp   short pix_op_mix_w_x
4692                              <1>
4693                              <1> pix_op_mix_w_x:
4694                              <1> pix_op_add_w_x:
4695                              <1> pix_op_sub_w_x:
4696                              <1> pix_op_rpl_w_x:
4697                              <1> pix_op_orc_w_x:
4698                              <1> pix_op_and_w_x:
4699                              <1> pix_op_xor_w_x:
4700                              <1>       ; 27/02/2021
4701                              <1>       ; 31/01/2021
4702                              <1>       ; ecx = bytes per row (to be applied)
4703                              <1>       ; edx = windows (screen) width in bytes
4704                              <1>       ; ebx = row count
4705                              <1>       ; eax = color
4706                              <1>       ; ebp = pixel operation subroutine address
4707 0000EA99 52                  <1>       push   edx
4708 0000EA9A 51                  <1>       push   ecx
4709 0000EA9B 57                  <1>       push   edi
4710 0000EA9C 8B0D[9E120300]      <1>       mov    ecx, [pixcount] ; 27/02/2021
4711 0000EAA2 FFD5                <1>       call   ebp ; call pixel-row operation
4712 0000EAA4 5F                  <1>       pop    edi
4713 0000EAA5 59                  <1>       pop    ecx ; bytes per row
4714 0000EAA6 010D[64030300]      <1>       add    [u.r0], ecx
4715 0000EAAC 5A                  <1>       pop    edx
4716 0000EAAD 01D7                <1>       add    edi, edx ; next row
4717 0000EAAF 4B                  <1>       dec    ebx
4718 0000EAB0 75E7                <1>       jnz    short pix_op_mix_w_x
4719 0000EAB2 C3                  <1>       retn
4720                              <1>
4721                              <1> pix_op_rpl:
4722                              <1>       ; 01/02/2021
4723                              <1>       ; REPLACE COLOR
4724                              <1>       ;
4725                              <1>       ; INPUT:
4726                              <1>       ;   CL = old/current color (8 bit, 256 colors)
4727                              <1>       ;  ECX = old/current color (16 bit and true colors)
4728                              <1>       ;   DL = new color (8 bit, 256 colors)
4729                              <1>       ;  EDX = new color (16 bit and true colors)
4730                              <1>       ;  ESI = start position (row, column)
4731                              <1>       ;        (HW = row, DX = column)
4732                              <1>       ;  EDI = size (rows, colums)
4733                              <1>       ;        (HW = rows, SI = columns)
4734                              <1>       ; OUTPUT:
4735                              <1>       ;     [u.r0] will be > 0 if succesful
4736                              <1>
4737 0000EAB3 F605[88120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
4738 0000EABA 754D                <1>       jnz    short pix_op_rpl_w ; window
4739                              <1>
4740 0000EABC 8B3D[8A120300]      <1>       mov    edi, [v_mem]
4741 0000EAC2 89FE                <1>       mov    esi, edi
4742                              <1>       ; ecx = old color (CL, CX, ECX) -to be replaced with-
4743                              <1>       ; edx = new color (CL, CX, ECX) -new one-
4744 0000EAC4 89D0                <1>       mov    eax, edx ; new color
4745 0000EAC6 890D[9A120300]      <1>       mov    [maskcolor], ecx ; old color
4746 0000EACC 8B0D[8E120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
4747                              <1> pix_op_rpl_0:
4748 0000EAD2 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
4749 0000EAD9 7707                <1>       ja     short pix_op_rpl_1
4750                              <1>
4751                              <1>       ; 256 colors (8bpp)
4752 0000EADB E8300A0000          <1>       call   pix_op_rpl_8
4753 0000EAE0 EB1E                <1>       jmp    short pix_op_rpl_4
4754                              <1>
4755                              <1> pix_op_rpl_1:
4756 0000EAE2 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
4757 0000EAE9 7710                <1>       ja     short pix_op_rpl_3 ; 32bpp
4758 0000EAEB 7207                <1>       jb     short pix_op_rpl_2 ; 16bpp
4759                              <1>
4760                              <1>       ; 24 bit true colors
```

```
0000EAED E8410A0000          <1>       call   pix_op_rpl_24
0000EAF2 EB0C                <1>       jmp    short pix_op_rpl_4
                             <1>
                             <1>       ; 65536 colors (16bpp)
                             <1> pix_op_rpl_2:
0000EAF4 E8270A0000          <1>       call   pix_op_rpl_16
0000EAF9 EB05                <1>       jmp    short pix_op_rpl_4
                             <1>
                             <1>       ; 32 bit true colors
                             <1> pix_op_rpl_3:
0000EAFB E8550A0000          <1>       call   pix_op_rpl_32
                             <1> pix_op_rpl_4:
0000EB00 29F7                <1>       sub    edi, esi
0000EB02 893D[64030300]      <1>       mov    [u.r0], edi
                             <1> pix_op_rpl_5:
0000EB08 C3                  <1>       retn
                             <1>
                             <1> pix_op_rpl_w:
                             <1>       ; 01/02/2021
0000EB09 890D[9A120300]      <1>       mov    [maskcolor], ecx ; old color
0000EB0F 52                  <1>       push   edx ; * ; new color
0000EB10 89F1                <1>       mov    ecx, esi ; win start pos
0000EB12 89FA                <1>       mov    edx, edi ; size (rows, cols)
0000EB14 E8FFFBFFFF          <1>       call   sysvideo_15_12 ; window preparations
0000EB19 58                  <1>       pop    eax ; * ; new color
0000EB1A 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>
0000EB1C 8B3D[92120300]      <1>       mov    edi, [v_str]
                             <1>
0000EB22 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EB29 7707                <1>       ja     short pix_op_rpl_w_1
                             <1>
                             <1>       ; 256 colors (8bpp)
0000EB2B BD[10F50000]        <1>       mov    ebp, pix_op_rpl_8
0000EB30 EB1E                <1>       jmp    short pix_op_rpl_w_4
                             <1>
                             <1> pix_op_rpl_w_1:
0000EB32 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EB39 7710                <1>       ja     short pix_op_rpl_w_3 ; 32bpp
0000EB3B 7207                <1>       jb     short pix_op_rpl_w_2 ; 16bpp
                             <1>
                             <1>       ; 24 bit true colors
0000EB3D BD[33F50000]        <1>       mov    ebp, pix_op_rpl_24
0000EB42 EB0C                <1>       jmp    short pix_op_rpl_w_4
                             <1>
                             <1>       ; 65536 colors (16bpp)
                             <1> pix_op_rpl_w_2:
0000EB44 BD[20F50000]        <1>       mov    ebp, pix_op_rpl_16
0000EB49 EB05                <1>       jmp    short pix_op_rpl_w_4
                             <1>
                             <1>       ; 32 bit true colors
                             <1> pix_op_rpl_w_3:
0000EB4B BD[55F50000]        <1>       mov    ebp, pix_op_rpl_32
                             <1> pix_op_rpl_w_4:
0000EB50 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>
0000EB55 F605[88120300]10    <1>       test   byte [v_ops], 10h ; display page or window ?
0000EB5C 7555                <1>       jnz    short pix_op_or_w ; window
                             <1>
0000EB5E 8B3D[8A120300]      <1>       mov    edi, [v_mem]
0000EB64 89FE                <1>       mov    esi, edi
                             <1>       ; ecx = color (CL, CX, ECX)
0000EB66 89C8                <1>       mov    eax, ecx
0000EB68 8B0D[8E120300]      <1>       mov    ecx, [v_siz] ; display page pixel count
                             <1>
0000EB6E F605[88120300]20    <1>       test   byte [v_ops], 20h ; masked color 'or' ?
0000EB75 7405                <1>       jz     short pix_op_or_0 ; no
0000EB77 E948110000          <1>       jmp    m_pix_op_or ; 'or' color except mask color
                             <1> pix_op_or_0:
0000EB7C 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EB83 7707                <1>       ja     short pix_op_or_1
                             <1>
                             <1>       ; 256 colors (8bpp)
0000EB85 E81C080000          <1>       call   pix_op_or_8
0000EB8A EB1E                <1>       jmp    short pix_op_or_4
                             <1>
                             <1> pix_op_or_1:
0000EB8C 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EB93 7710                <1>       ja     short pix_op_or_3 ; 32bpp
0000EB95 7207                <1>       jb     short pix_op_or_2 ; 16bpp
```

```
                                <1>
                                <1>       ; 24 bit true colors
0000EB97 E818080000             <1>       call   pix_op_or_24
0000EB9C EB0C                   <1>       jmp    short pix_op_or_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_or_2:
0000EB9E E809080000             <1>       call   pix_op_or_16
0000EBA3 EB05                   <1>       jmp    short pix_op_or_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_or_3:
0000EBA5 E819080000             <1>       call   pix_op_or_32
                                <1> pix_op_or_4:
0000EBAA 29F7                   <1>       sub    edi, esi
0000EBAC 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_or_5:
0000EBB2 C3                     <1>       retn
                                <1>
                                <1> pix_op_or_w:
                                <1>       ; 31/01/2021
0000EBB3 51                     <1>       push   ecx ; * ; color
0000EBB4 89D1                   <1>       mov    ecx, edx ; win start pos
0000EBB6 89F2                   <1>       mov    edx, esi ; size (rows, cols)
0000EBB8 E85BFBFFFF             <1>       call   sysvideo_15_12 ; window preparations
0000EBBD 58                     <1>       pop    eax ; * ; color
0000EBBE 72F2                   <1>       jc     short pix_op_or_5
                                <1>
0000EBC0 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color 'or' ?
0000EBC7 7405                   <1>       jz     short pix_op_or_w_0 ; no
0000EBC9 E983110000             <1>       jmp    m_pix_op_or_w
                                <1>                      ; window 'or' color except mask color
                                <1> pix_op_or_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>
0000EBCE 8B3D[92120300]         <1>       mov    edi, [v_str]
0000EBD4 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EBDB 7707                   <1>       ja     short pix_op_or_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EBDD BD[A6F30000]           <1>       mov    ebp, pix_op_or_8
0000EBE2 EB1E                   <1>       jmp    short pix_op_or_w_4
                                <1>
                                <1> pix_op_or_w_1:
0000EBE4 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EBEB 7710                   <1>       ja     short pix_op_or_w_3 ; 32bpp
0000EBED 7207                   <1>       jb     short pix_op_or_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EBEF BD[B4F30000]           <1>       mov    ebp, pix_op_or_24
0000EBF4 EB0C                   <1>       jmp    short pix_op_or_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_or_w_2:
0000EBF6 BD[ACF30000]           <1>       mov    ebp, pix_op_or_16
0000EBFB EB05                   <1>       jmp    short pix_op_or_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_or_w_3:
0000EBFD BD[C3F30000]           <1>       mov    ebp, pix_op_or_32
                                <1> pix_op_or_w_4:
0000EC02 E992FEFFFF             <1>       jmp    pix_op_orc_w_x
                                <1>
                                <1> pix_op_and:
                                <1>       ; 31/01/2021
                                <1>       ; AND 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>
0000EC07 F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000EC0E 7555                   <1>       jnz    short pix_op_and_w ; window
                                <1>
0000EC10 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000EC16 89FE                   <1>       mov    esi, edi
                                <1>       ; ecx = color (CL, CX, ECX)
0000EC18 89C8                   <1>       mov    eax, ecx
0000EC1A 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000EC20 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color 'and' ?
0000EC27 7405                   <1>       jz     short pix_op_and_0 ; no
0000EC29 E9D60F0000             <1>       jmp    m_pix_op_and ; 'and' color except mask color
                                <1> pix_op_and_0:
0000EC2E 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EC35 7707                   <1>       ja     short pix_op_and_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EC37 E88F070000             <1>       call   pix_op_and_8
0000EC3C EB1E                   <1>       jmp    short pix_op_and_4
                                <1>
                                <1> pix_op_and_1:
0000EC3E 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
```

```
4971 0000EC45 7710              <1>       ja     short pix_op_and_3 ; 32bpp
4972 0000EC47 7207              <1>       jb     short pix_op_and_2 ; 16bpp
4973                            <1>
4974                            <1>       ; 24 bit true colors
4975 0000EC49 E88B070000        <1>       call   pix_op_and_24
4976 0000EC4E EB0C              <1>       jmp    short pix_op_and_4
4977                            <1>
4978                            <1>       ; 65536 colors (16bpp)
4979                            <1> pix_op_and_2:
4980 0000EC50 E87C070000        <1>       call   pix_op_and_16
4981 0000EC55 EB05              <1>       jmp    short pix_op_and_4
4982                            <1>
4983                            <1>       ; 32 bit true colors
4984                            <1> pix_op_and_3:
4985 0000EC57 E88C070000        <1>       call   pix_op_and_32
4986                            <1> pix_op_and_4:
4987 0000EC5C 29F7              <1>       sub    edi, esi
4988 0000EC5E 893D[64030300]    <1>       mov    [u.r0], edi
4989                            <1> pix_op_and_5:
4990 0000EC64 C3                <1>       retn
4991                            <1>
4992                            <1> pix_op_and_w:
4993                            <1>       ; 31/01/2021
4994 0000EC65 51                <1>       push   ecx ; * ; color
4995 0000EC66 89D1              <1>       mov    ecx, edx ; win start pos
4996 0000EC68 89F2              <1>       mov    edx, esi ; size (rows, cols)
4997 0000EC6A E8A9FAFFFF        <1>       call   sysvideo_15_12 ; window preparations
4998 0000EC6F 58                <1>       pop    eax ; * ; color
4999 0000EC70 72F2              <1>       jc     short pix_op_and_5
5000                            <1>
5001 0000EC72 F605[88120300]20  <1>       test   byte [v_ops], 20h ; masked color 'and' ?
5002 0000EC79 7405              <1>       jz     short pix_op_and_w_0 ; no
5003 0000EC7B E911100000        <1>       jmp    m_pix_op_and_w
5004                            <1>                    ; window 'and' color except mask color
5005                            <1> pix_op_and_w_0:
5006                            <1>       ; ecx = bytes per row (to be applied)
5007                            <1>       ; edx = screen width in bytes
5008                            <1>       ; ebx = row count
5009                            <1>       ; eax = color
5010                            <1>
5011 0000EC80 8B3D[92120300]    <1>       mov    edi, [v_str]
5012 0000EC86 803D[89120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
5013 0000EC8D 7707              <1>       ja     short pix_op_and_w_1
5014                            <1>
5015                            <1>       ; 256 colors (8bpp)
5016 0000EC8F BD[CBF30000]      <1>       mov    ebp, pix_op_and_8
5017 0000EC94 EB1E              <1>       jmp    short pix_op_and_w_4
5018                            <1>
5019                            <1> pix_op_and_w_1:
5020 0000EC96 803D[89120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
5021 0000EC9D 7710              <1>       ja     short pix_op_and_w_3 ; 32bpp
5022 0000EC9F 7207              <1>       jb     short pix_op_and_w_2 ; 16bpp
5023                            <1>
5024                            <1>       ; 24 bit true colors
5025 0000ECA1 BD[D9F30000]      <1>       mov    ebp, pix_op_and_24
5026 0000ECA6 EB0C              <1>       jmp    short pix_op_and_w_4
5027                            <1>
5028                            <1>       ; 65536 colors (16bpp)
5029                            <1> pix_op_and_w_2:
5030 0000ECA8 BD[D1F30000]      <1>       mov    ebp, pix_op_and_16
5031 0000ECAD EB05              <1>       jmp    short pix_op_and_w_4
5032                            <1>
5033                            <1>       ; 32 bit true colors
5034                            <1> pix_op_and_w_3:
5035 0000ECAF BD[E8F30000]      <1>       mov    ebp, pix_op_and_32
5036                            <1> pix_op_and_w_4:
5037 0000ECB4 E9E0FDFFFF        <1>       jmp    pix_op_and_w_x
5038                            <1>
5039                            <1> pix_op_xor:
5040                            <1>       ; 31/01/2021
5041                            <1>       ; XOR COLOR
5042                            <1>       ;
5043                            <1>       ; INPUT:
5044                            <1>       ;   CL = color (8 bit, 256 colors)
5045                            <1>       ;  ECX = color (16 bit and true colors)
5046                            <1>       ;  EDX = start position (row, column)
5047                            <1>       ;        (HW = row, DX = column)
5048                            <1>       ;  ESI = size (rows, colums)
5049                            <1>       ;        (HW = rows, SI = columns)
5050                            <1>       ;
5051                            <1>       ;  [maskcolor] = mask color (to be excluded)
5052                            <1>       ;
5053                            <1>       ; OUTPUT:
5054                            <1>       ;      [u.r0] will be > 0 if succesful
5055                            <1>
5056 0000ECB9 F605[88120300]10  <1>       test   byte [v_ops], 10h ; display page or window ?
5057 0000ECC0 7555              <1>       jnz    short pix_op_xor_w ; window
5058                            <1>
5059 0000ECC2 8B3D[8A120300]    <1>       mov    edi, [v_mem]
5060 0000ECC8 89FE              <1>       mov    esi, edi
5061                            <1>       ; ecx = color (CL, CX, ECX)
5062 0000ECCA 89C8              <1>       mov    eax, ecx
5063 0000ECCC 8B0D[8E120300]    <1>       mov    ecx, [v_siz] ; display page pixel count
5064                            <1>
5065 0000ECD2 F605[88120300]20  <1>       test   byte [v_ops], 20h ; masked color 'xor' ?
5066 0000ECD9 7405              <1>       jz     short pix_op_xor_0 ; no
5067 0000ECDB E9A4100000        <1>       jmp    m_pix_op_xor ; 'xor' color except mask color
5068                            <1> pix_op_xor_0:
5069 0000ECE0 803D[89120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
5070 0000ECE7 7707              <1>       ja     short pix_op_xor_1
5071                            <1>
5072                            <1>       ; 256 colors (8bpp)
5073 0000ECE9 E802070000        <1>       call   pix_op_xor_8
5074 0000ECEE EB1E              <1>       jmp    short pix_op_xor_4
5075                            <1>
```

```
                                <1> pix_op_xor_1:
0000ECF0 803D[89120300]18       <1>         cmp   byte [v_bpp], 24 ; 24bpp
0000ECF7 7710                   <1>         ja    short pix_op_xor_3 ; 32bpp
0000ECF9 7207                   <1>         jb    short pix_op_xor_2 ; 16bpp
                                <1>
                                <1>         ; 24 bit true colors
0000ECFB E8FE060000             <1>         call  pix_op_xor_24
0000ED00 EB0C                   <1>         jmp   short pix_op_xor_4
                                <1>
                                <1>         ; 65536 colors (16bpp)
                                <1> pix_op_xor_2:
0000ED02 E8EF060000             <1>         call  pix_op_xor_16
0000ED07 EB05                   <1>         jmp   short pix_op_xor_4
                                <1>
                                <1>         ; 32 bit true colors
                                <1> pix_op_xor_3:
0000ED09 E8FF060000             <1>         call  pix_op_xor_32
                                <1> pix_op_xor_4:
0000ED0E 29F7                   <1>         sub   edi, esi
0000ED10 893D[64030300]         <1>         mov   [u.r0], edi
                                <1> pix_op_xor_5:
0000ED16 C3                     <1>         retn
                                <1>
                                <1> pix_op_xor_w:
                                <1>         ; 31/01/2021
0000ED17 51                     <1>         push  ecx ; * ; color
0000ED18 89D1                   <1>         mov   ecx, edx ; win start pos
0000ED1A 89F2                   <1>         mov   edx, esi ; size (rows, cols)
0000ED1C E8F7F9FFFF             <1>         call  sysvideo_15_12 ; window preparations
0000ED21 58                     <1>         pop   eax ; * ; color
0000ED22 72F2                   <1>         jc    short pix_op_xor_5
                                <1>
0000ED24 F605[88120300]20       <1>         test  byte [v_ops], 20h ; masked color 'xor' ?
0000ED2B 7405                   <1>         jz    short pix_op_xor_w_0 ; no
0000ED2D E9DF100000             <1>         jmp   m_pix_op_xor_w
                                <1>                   ; window 'xor' color except mask color
                                <1> pix_op_xor_w_0:
                                <1>         ; ecx = bytes per row (to be applied)
                                <1>         ; edx = screen width in bytes
                                <1>         ; ebx = row count
                                <1>         ; eax = color
                                <1>
0000ED32 8B3D[92120300]         <1>         mov   edi, [v_str]
0000ED38 803D[89120300]08       <1>         cmp   byte [v_bpp], 8 ; 8bpp
0000ED3F 7707                   <1>         ja    short pix_op_xor_w_1
                                <1>
                                <1>         ; 256 colors (8bpp)
0000ED41 BD[F0F30000]           <1>         mov   ebp, pix_op_xor_8
0000ED46 EB1E                   <1>         jmp   short pix_op_xor_w_4
                                <1>
                                <1> pix_op_xor_w_1:
0000ED48 803D[89120300]18       <1>         cmp   byte [v_bpp], 24 ; 24bpp
0000ED4F 7710                   <1>         ja    short pix_op_xor_w_3 ; 32bpp
0000ED51 7207                   <1>         jb    short pix_op_xor_w_2 ; 16bpp
                                <1>
                                <1>         ; 24 bit true colors
0000ED53 BD[FEF30000]           <1>         mov   ebp, pix_op_xor_24
0000ED58 EB0C                   <1>         jmp   short pix_op_xor_w_4
                                <1>
                                <1>         ; 65536 colors (16bpp)
                                <1> pix_op_xor_w_2:
0000ED5A BD[F6F30000]           <1>         mov   ebp, pix_op_xor_16
0000ED5F EB05                   <1>         jmp   short pix_op_xor_w_4
                                <1>
                                <1>         ; 32 bit true colors
                                <1> pix_op_xor_w_3:
0000ED61 BD[0DF40000]           <1>         mov   ebp, pix_op_xor_32
                                <1> pix_op_xor_w_4:
0000ED66 E92EFDFFFF             <1>         jmp   pix_op_xor_w_x
                                <1>
                                <1> pix_op_new:
                                <1>         ; 31/01/2021
                                <1>         ; 30/01/2021
                                <1>         ; CHANGE 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>
0000ED6B F605[88120300]10       <1>         test  byte [v_ops], 10h ; display page or window ?
0000ED72 7554                   <1>         jnz   short pix_op_new_w ; window
                                <1>
0000ED74 8B3D[8A120300]         <1>         mov   edi, [v_mem]
0000ED7A 89FE                   <1>         mov   esi, edi
                                <1>         ; ecx = color (CL, CX, ECX)
0000ED7C 89C8                   <1>         mov   eax, ecx
0000ED7E 8B0D[8E120300]         <1>         mov   ecx, [v_siz] ; display page pixel count
                                <1>
0000ED84 F605[88120300]20       <1>         test  byte [v_ops], 20h ; masked color change ?
0000ED8B 7405                   <1>         jz    short pix_op_new_0 ; no
0000ED8D E90D0B0000             <1>         jmp   m_pix_op_new ; change color except mask color
                                <1> pix_op_new_0:
0000ED92 803D[89120300]08       <1>         cmp   byte [v_bpp], 8 ; 8bpp
0000ED99 7706                   <1>         ja    short pix_op_new_2
                                <1>
                                <1>         ; 256 colors (8bpp)
```

```
                                <1> pix_op_new_1:
0000ED9B 88C4                   <1>       mov    ah, al
0000ED9D D1E9                   <1>       shr    ecx, 1
0000ED9F EB12                   <1>       jmp    short pix_op_new_3
                                <1>
                                <1> pix_op_new_2:
0000EDA1 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EDA8 7713                   <1>       ja     short pix_op_new_4 ; 32bpp
0000EDAA 7207                   <1>       jb     short pix_op_new_3 ; 16bpp
                                <1>
                                <1>       ; 31/01/2021
                                <1>
                                <1>       ; 24 bit true colors
0000EDAC E84A050000             <1>       call   pix_op_new_24
                                <1>
0000EDB1 EB0C                   <1>       jmp    short pix_op_new_5
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_new_3:
0000EDB3 89C2                   <1>       mov    edx, eax
0000EDB5 C1E010                 <1>       shl    eax, 16
0000EDB8 6689D0                 <1>       mov    ax, dx
0000EDBB D1E9                   <1>       shr    ecx, 1 ; dword counts
                                <1>       ; 32 bit true colors
                                <1> pix_op_new_4:
0000EDBD F3AB                   <1>       rep    stosd
                                <1> pix_op_new_5:
0000EDBF 29F7                   <1>       sub    edi, esi
0000EDC1 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_new_6:
0000EDC7 C3                     <1>       retn
                                <1>
                                <1> pix_op_new_w:
                                <1>       ; 31/01/2021
                                <1>       ; 30/01/2021
0000EDC8 51                     <1>       push   ecx ; * ; color
0000EDC9 89D1                   <1>       mov    ecx, edx ; win start pos
0000EDCB 89F2                   <1>       mov    edx, esi ; size (rows, cols)
0000EDCD E846F9FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000EDD2 58                     <1>       pop    eax ; * ; color
0000EDD3 72F2                   <1>       jc     short pix_op_new_6
                                <1>
0000EDD5 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked color change ?
0000EDDC 7405                   <1>       jz     short pix_op_new_w_0 ; no
0000EDDE E94A0B0000             <1>       jmp    m_pix_op_new_w
                                <1>                      ; window chg color except mask color
                                <1> pix_op_new_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>
0000EDE3 8B3D[92120300]         <1>       mov    edi, [v_str]
                                <1>
0000EDE9 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EDF0 7707                   <1>       ja     short pix_op_new_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EDF2 BD[F4F20000]           <1>       mov    ebp, pix_op_new_8
0000EDF7 EB1E                   <1>       jmp    short pix_op_new_w_x
                                <1>
                                <1> pix_op_new_w_1:
0000EDF9 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EE00 7710                   <1>       ja     short pix_op_new_w_3 ; 32bpp
0000EE02 7207                   <1>       jb     short pix_op_new_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EE04 BD[FBF20000]           <1>       mov    ebp, pix_op_new_24
0000EE09 EB0C                   <1>       jmp    short pix_op_new_w_x
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_new_w_2:
0000EE0B BD[F7F20000]           <1>       mov    ebp, pix_op_new_16
0000EE10 EB05                   <1>       jmp    short pix_op_new_w_x
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_new_w_3:
0000EE12 BD[07F30000]           <1>       mov    ebp, pix_op_new_32
                                <1>       ;jmp   short pix_op_new_w_x
                                <1>
                                <1> pix_op_new_w_x:
                                <1> pix_op_not_w_x:
                                <1> pix_op_neg_w_x:
                                <1> pix_op_inc_w_x:
                                <1> pix_op_dec_w_x:
                                <1>       ; 27/02/2021
                                <1>       ; 01/02/2021
                                <1>       ; 31/01/2021
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = windows (screen) width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>       ; ebp = pixel operation subroutine address
                                <1>       ;push  edx ; 01/02/2021
0000EE17 51                     <1>       push   ecx
0000EE18 57                     <1>       push   edi
0000EE19 8B0D[9E120300]         <1>       mov    ecx, [pixcount] ; 27/02/2021
0000EE1F FFD5                   <1>       call   ebp ; call pixel-row operation
0000EE21 5F                     <1>       pop    edi
0000EE22 59                     <1>       pop    ecx ; bytes per row
0000EE23 010D[64030300]         <1>       add    [u.r0], ecx
                                <1>       ;pop   edx ; 01/02/2021
0000EE29 01D7                   <1>       add    edi, edx ; next row
0000EE2B 4B                     <1>       dec    ebx
0000EE2C 75E9                   <1>       jnz    short pix_op_new_w_x
```

```
5286 0000EE2E C3                  <1>       retn
5287                              <1>
5288                              <1> pix_op_not:
5289                              <1>       ; 31/01/2021
5290                              <1>       ; NOT COLOR
5291                              <1>       ;
5292                              <1>       ; INPUT:
5293                              <1>       ;  ECX = start position (row, column)
5294                              <1>       ;         (HW = row, CX = column)
5295                              <1>       ;  EDX = size (rows, colums)
5296                              <1>       ;         (HW = rows, DX = columns)
5297                              <1>       ;       (0 -> invalid
5298                              <1>       ;        (1 -> horizontal or vertical line)
5299                              <1>       ;  [maskcolor] = mask color (to be excluded)
5300                              <1>       ;
5301                              <1>       ; OUTPUT:
5302                              <1>       ;      [u.r0] will be > 0 if succesful
5303                              <1>
5304 0000EE2F F605[88120300]10   <1>       test   byte [v_ops], 10h ; display page or window ?
5305 0000EE36 7553               <1>       jnz    short pix_op_not_w ; window
5306                              <1>
5307 0000EE38 8B3D[8A120300]     <1>       mov    edi, [v_mem]
5308 0000EE3E 89FE               <1>       mov    esi, edi
5309 0000EE40 8B0D[8E120300]     <1>       mov    ecx, [v_siz] ; display page pixel count
5310                              <1>
5311 0000EE46 F605[88120300]20   <1>       test   byte [v_ops], 20h ; masked color 'not' ?
5312 0000EE4D 7405               <1>       jz     short pix_op_not_0 ; no
5313 0000EE4F E9F00F0000         <1>       jmp    m_pix_op_not ; 'not' color except mask color
5314                              <1> pix_op_not_0:
5315 0000EE54 803D[89120300]08   <1>       cmp    byte [v_bpp], 8 ; 8bpp
5316 0000EE5B 7707               <1>       ja     short pix_op_not_1
5317                              <1>
5318                              <1>       ; 256 colors (8bpp)
5319 0000EE5D E8F6050000         <1>       call   pix_op_not_8
5320 0000EE62 EB1E               <1>       jmp    short pix_op_not_4
5321                              <1>
5322                              <1> pix_op_not_1:
5323 0000EE64 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
5324 0000EE6B 7710               <1>       ja     short pix_op_not_3 ; 32bpp
5325 0000EE6D 7207               <1>       jb     short pix_op_not_2 ; 16bpp
5326                              <1>
5327                              <1>       ; 24 bit true colors
5328 0000EE6F E8F2050000         <1>       call   pix_op_not_24
5329 0000EE74 EB0C               <1>       jmp    short pix_op_not_4
5330                              <1>
5331                              <1>       ; 65536 colors (16bpp)
5332                              <1> pix_op_not_2:
5333 0000EE76 E8E3050000         <1>       call   pix_op_not_16
5334 0000EE7B EB05               <1>       jmp    short pix_op_not_4
5335                              <1>
5336                              <1>       ; 32 bit true colors
5337                              <1> pix_op_not_3:
5338 0000EE7D E8EF050000         <1>       call   pix_op_not_32
5339                              <1> pix_op_not_4:
5340 0000EE82 29F7               <1>       sub    edi, esi
5341 0000EE84 893D[64030300]     <1>       mov    [u.r0], edi
5342                              <1> pix_op_not_5:
5343 0000EE8A C3                 <1>       retn
5344                              <1>
5345                              <1> pix_op_not_w:
5346                              <1>       ; 31/01/2021
5347                              <1>       ; ecx = win start pos (row, column)
5348                              <1>       ; edx = size (rows, columns)
5349 0000EE8B E888F8FFFF         <1>       call   sysvideo_15_12 ; window preparations
5350 0000EE90 72F8               <1>       jc     short pix_op_not_5
5351                              <1>
5352 0000EE92 F605[88120300]20   <1>       test   byte [v_ops], 20h ; masked color 'not' ?
5353 0000EE99 7405               <1>       jz     short pix_op_not_w_0 ; no
5354 0000EE9B E929100000         <1>       jmp    m_pix_op_not_w
5355                              <1>                      ; window 'not' color except mask color
5356                              <1> pix_op_not_w_0:
5357                              <1>       ; ecx = bytes per row (to be applied)
5358                              <1>       ; edx = screen width in bytes
5359                              <1>       ; ebx = row count
5360                              <1>
5361 0000EEA0 8B3D[92120300]     <1>       mov    edi, [v_str]
5362                              <1>
5363 0000EEA6 803D[89120300]08   <1>       cmp    byte [v_bpp], 8 ; 8bpp
5364 0000EEAD 7707               <1>       ja     short pix_op_not_w_1
5365                              <1>
5366                              <1>       ; 256 colors (8bpp)
5367 0000EEAF BD[58F40000]       <1>       mov    ebp, pix_op_not_8
5368 0000EEB4 EB1E               <1>       jmp    short pix_op_not_w_4
5369                              <1>
5370                              <1> pix_op_not_w_1:
5371 0000EEB6 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
5372 0000EEBD 7710               <1>       ja     short pix_op_not_w_3 ; 32bpp
5373 0000EEBF 7207               <1>       jb     short pix_op_not_w_2 ; 16bpp
5374                              <1>
5375                              <1>       ; 24 bit true colors
5376 0000EEC1 BD[66F40000]       <1>       mov    ebp, pix_op_not_24
5377 0000EEC6 EB0C               <1>       jmp    short pix_op_not_w_4
5378                              <1>
5379                              <1>       ; 65536 colors (16bpp)
5380                              <1> pix_op_not_w_2:
5381 0000EEC8 BD[5EF40000]       <1>       mov    ebp, pix_op_not_16
5382 0000EECD EB05               <1>       jmp    short pix_op_not_w_4
5383                              <1>
5384                              <1>       ; 32 bit true colors
5385                              <1> pix_op_not_w_3:
5386 0000EECF BD[71F40000]       <1>       mov    ebp, pix_op_not_32
5387                              <1> pix_op_not_w_4:
5388 0000EED4 E93EFFFFFF         <1>       jmp    pix_op_not_w_x
5389                              <1>
5390                              <1> pix_op_neg:
```

```
                                <1>       ; 31/01/2021
                                <1>       ; NEGATE COLOR
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;  ECX = start position (row, column)
                                <1>       ;         (HW = row, CX = column)
                                <1>       ;  EDX = size (rows, colums)
                                <1>       ;         (HW = rows, DX = columns)
                                <1>       ;       (0 -> invalid
                                <1>       ;         (1 -> horizontal or vertical line)
                                <1>       ;  [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
0000EED9 F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000EEE0 7553                   <1>       jnz    short pix_op_neg_w ; window
                                <1>
0000EEE2 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000EEE8 89FE                   <1>       mov    esi, edi
0000EEEA 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000EEF0 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked negate color ?
0000EEF7 7405                   <1>       jz     short pix_op_neg_0 ; no
0000EEF9 E9FE0F0000             <1>       jmp    m_pix_op_neg ; 'neg' color except mask color
                                <1> pix_op_neg_0:
0000EEFE 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EF05 7707                   <1>       ja     short pix_op_neg_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EF07 E86D050000             <1>       call   pix_op_neg_8
0000EF0C EB1E                   <1>       jmp    short pix_op_neg_4
                                <1>
                                <1> pix_op_neg_1:
0000EF0E 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EF15 7710                   <1>       ja     short pix_op_neg_3 ; 32bpp
0000EF17 7207                   <1>       jb     short pix_op_neg_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EF19 E869050000             <1>       call   pix_op_neg_24
0000EF1E EB0C                   <1>       jmp    short pix_op_neg_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_neg_2:
0000EF20 E85A050000             <1>       call   pix_op_neg_16
0000EF25 EB05                   <1>       jmp    short pix_op_neg_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_neg_3:
0000EF27 E86D050000             <1>       call   pix_op_neg_32
                                <1> pix_op_neg_4:
0000EF2C 29F7                   <1>       sub    edi, esi
0000EF2E 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_neg_5:
0000EF34 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)
0000EF35 E8DEF7FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000EF3A 72F8                   <1>       jc     short pix_op_neg_5
                                <1>
0000EF3C F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked negate color ?
0000EF43 7405                   <1>       jz     short pix_op_neg_w_0 ; no
0000EF45 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>
0000EF4A 8B3D[92120300]         <1>       mov    edi, [v_str]
                                <1>
0000EF50 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EF57 7707                   <1>       ja     short pix_op_neg_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EF59 BD[79F40000]           <1>       mov    ebp, pix_op_neg_8
0000EF5E EB1E                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1> pix_op_neg_w_1:
0000EF60 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EF67 7710                   <1>       ja     short pix_op_neg_w_3 ; 32bpp
0000EF69 7207                   <1>       jb     short pix_op_neg_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EF6B BD[87F40000]           <1>       mov    ebp, pix_op_neg_24
0000EF70 EB0C                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_neg_w_2:
0000EF72 BD[7FF40000]           <1>       mov    ebp, pix_op_neg_16
0000EF77 EB05                   <1>       jmp    short pix_op_neg_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_neg_w_3:
0000EF79 BD[99F40000]           <1>       mov    ebp, pix_op_neg_32
                                <1> pix_op_neg_w_4:
0000EF7E E994FEFFFF             <1>       jmp    pix_op_neg_w_x
                                <1>
                                <1> pix_op_inc:
                                <1>       ; 31/01/2021
                                <1>       ; INCREASE COLOR
                                <1>       ;
```

```
                                <1>       ; INPUT:
                                <1>       ;  ECX = start position (row, column)
                                <1>       ;         (HW = row, CX = column)
                                <1>       ;  EDX = size (rows, colums)
                                <1>       ;         (HW = rows, DX = columns)
                                <1>       ;       (0 -> invalid
                                <1>       ;         (1 -> horizontal or vertical line)
                                <1>       ;  [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
0000EF83 F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000EF8A 7553                   <1>       jnz    short pix_op_inc_w ; window
                                <1>
0000EF8C 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000EF92 89FE                   <1>       mov    esi, edi
0000EF94 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000EF9A F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked increase color ?
0000EFA1 7405                   <1>       jz     short pix_op_inc_0 ; no
0000EFA3 E90C100000             <1>       jmp    m_pix_op_inc ; 'inc' color except mask color
                                <1> pix_op_inc_0:
0000EFA8 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000EFAF 7707                   <1>       ja     short pix_op_inc_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000EFB1 E8EB040000             <1>       call   pix_op_inc_8
0000EFB6 EB1E                   <1>       jmp    short pix_op_inc_4
                                <1>
                                <1> pix_op_inc_1:
0000EFB8 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000EFBF 7710                   <1>       ja     short pix_op_inc_3 ; 32bpp
0000EFC1 7207                   <1>       jb     short pix_op_inc_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000EFC3 E8F0040000             <1>       call   pix_op_inc_24
0000EFC8 EB0C                   <1>       jmp    short pix_op_inc_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_inc_2:
0000EFCA E8DC040000             <1>       call   pix_op_inc_16
0000EFCF EB05                   <1>       jmp    short pix_op_inc_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_inc_3:
0000EFD1 E8F6040000             <1>       call   pix_op_inc_32
                                <1> pix_op_inc_4:
0000EFD6 29F7                   <1>       sub    edi, esi
0000EFD8 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_inc_5:
0000EFDE C3                     <1>       retn
                                <1>
                                <1> pix_op_inc_w:
                                <1>       ; 31/01/2021
                                <1>       ; ecx = win start pos (row, column)
                                <1>       ; edx = size (rows, columns)
0000EFDF E834F7FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000EFE4 72F8                   <1>       jc     short pix_op_inc_5
                                <1>
0000EFE6 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked increase color ?
0000EFED 7405                   <1>       jz     short pix_op_inc_w_0 ; no
0000EFEF E959100000             <1>       jmp    m_pix_op_inc_w
                                <1>                      ; window 'inc' color except mask color
                                <1> pix_op_inc_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>
0000EFF4 8B3D[92120300]         <1>       mov    edi, [v_str]
                                <1>
0000EFFA 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000F001 7707                   <1>       ja     short pix_op_inc_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000F003 BD[A1F40000]           <1>       mov    ebp, pix_op_inc_8
0000F008 EB1E                   <1>       jmp    short pix_op_inc_w_4
                                <1>
                                <1> pix_op_inc_w_1:
0000F00A 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000F011 7710                   <1>       ja     short pix_op_inc_w_3 ; 32bpp
0000F013 7207                   <1>       jb     short pix_op_inc_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000F015 BD[B8F40000]           <1>       mov    ebp, pix_op_inc_24
0000F01A EB0C                   <1>       jmp    short pix_op_inc_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_inc_w_2:
0000F01C BD[ABF40000]           <1>       mov    ebp, pix_op_inc_16
0000F021 EB05                   <1>       jmp    short pix_op_inc_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_inc_w_3:
0000F023 BD[CCF40000]           <1>       mov    ebp, pix_op_inc_32
                                <1> pix_op_inc_w_4:
0000F028 E9EAFDFFFF             <1>       jmp    pix_op_inc_w_x
                                <1>
                                <1> pix_op_dec:
                                <1>       ; 31/01/2021
                                <1>       ; DECREASE COLOR
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;  ECX = start position (row, column)
                                <1>       ;         (HW = row, CX = column)
```

```
                                <1>       ;  EDX = size (rows, colums)
                                <1>       ;         (HW = rows, DX = columns)
                                <1>       ;       (0 -> invalid
                                <1>       ;         (1 -> horizontal or vertical line)
                                <1>       ;  [maskcolor] = mask color (to be excluded)
                                <1>       ;
                                <1>       ; OUTPUT:
                                <1>       ;      [u.r0] will be > 0 if succesful
                                <1>
0000F02D F605[88120300]10       <1>       test   byte [v_ops], 10h ; display page or window ?
0000F034 7553                   <1>       jnz    short pix_op_dec_w ; window
                                <1>
0000F036 8B3D[8A120300]         <1>       mov    edi, [v_mem]
0000F03C 89FE                   <1>       mov    esi, edi
0000F03E 8B0D[8E120300]         <1>       mov    ecx, [v_siz] ; display page pixel count
                                <1>
0000F044 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked decrease color ?
0000F04B 7405                   <1>       jz     short pix_op_dec_0 ; no
0000F04D E92E100000             <1>       jmp    m_pix_op_dec ; 'dec' color except mask color
                                <1> pix_op_dec_0:
0000F052 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000F059 7707                   <1>       ja     short pix_op_dec_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000F05B E878040000             <1>       call   pix_op_dec_8
0000F060 EB1E                   <1>       jmp    short pix_op_dec_4
                                <1>
                                <1> pix_op_dec_1:
0000F062 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000F069 7710                   <1>       ja     short pix_op_dec_3 ; 32bpp
0000F06B 7207                   <1>       jb     short pix_op_dec_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000F06D E87D040000             <1>       call   pix_op_dec_24
0000F072 EB0C                   <1>       jmp    short pix_op_dec_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_dec_2:
0000F074 E869040000             <1>       call   pix_op_dec_16
0000F079 EB05                   <1>       jmp    short pix_op_dec_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_dec_3:
0000F07B E882040000             <1>       call   pix_op_dec_32
                                <1> pix_op_dec_4:
0000F080 29F7                   <1>       sub    edi, esi
0000F082 893D[64030300]         <1>       mov    [u.r0], edi
                                <1> pix_op_dec_5:
0000F088 C3                     <1>       retn
                                <1>
                                <1> pix_op_dec_w:
                                <1>       ; 31/01/2021
                                <1>       ; ecx = win start pos (row, column)
                                <1>       ; edx = size (rows, columns)
0000F089 E88AF6FFFF             <1>       call   sysvideo_15_12 ; window preparations
0000F08E 72F8                   <1>       jc     short pix_op_dec_5
                                <1>
0000F090 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked decrease color ?
0000F097 7405                   <1>       jz     short pix_op_dec_w_0 ; no
0000F099 E976100000             <1>       jmp    m_pix_op_dec_w
                                <1>                      ; window 'dec' color except mask color
                                <1> pix_op_dec_w_0:
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = screen width in bytes
                                <1>       ; ebx = row count
                                <1>
0000F09E 8B3D[92120300]         <1>       mov    edi, [v_str]
                                <1>
0000F0A4 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000F0AB 7707                   <1>       ja     short pix_op_dec_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000F0AD BD[D8F40000]           <1>       mov    ebp, pix_op_dec_8
0000F0B2 EB1E                   <1>       jmp    short pix_op_dec_w_4
                                <1>
                                <1> pix_op_dec_w_1:
0000F0B4 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000F0BB 7710                   <1>       ja     short pix_op_dec_w_3 ; 32bpp
0000F0BD 7207                   <1>       jb     short pix_op_dec_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000F0BF BD[EFF40000]           <1>       mov    ebp, pix_op_dec_24
0000F0C4 EB0C                   <1>       jmp    short pix_op_dec_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> pix_op_dec_w_2:
0000F0C6 BD[E2F40000]           <1>       mov    ebp, pix_op_dec_16
0000F0CB EB05                   <1>       jmp    short pix_op_dec_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> pix_op_dec_w_3:
0000F0CD BD[02F50000]           <1>       mov    ebp, pix_op_dec_32
                                <1> pix_op_dec_w_4:
0000F0D2 E940FDFFFF             <1>       jmp    pix_op_dec_w_x
                                <1>
                                <1> pix_op_blk:
                                <1>       ; 23/01/2021
                                <1>       ; 22/02/2021
                                <1>       ; 02/02/2021
                                <1>       ; COPY PIXEL BLOCK -system to system-
                                <1>       ; WRITE PIXEL BLOCKS -user to system-
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;   -If BL bit 5 is 0-
                                <1>       ;    ECX = start position (row, column) (*)
```

```
                                <1>       ;    (HW = row, CX = column)
                                <1>       ;    EDX = size (rows, colums) (*)
                                <1>       ;    (HW = rows, DX = columns)
                                <1>       ;         (0 -> invalid)
                                <1>       ;         (1 -> horizontal or vertical line)
                                <1>       ;    ESI = destination (row, column) (***)
                                <1>       ;   -If BL bit 5 is 1-
                                <1>       ;     CL = color (8 bit, 256 colors)
                                <1>       ;    ECX = color (16 bit and true colors)
                                <1>       ;    EDX = count of blocks (not bytes)
                                <1>       ;       (limit: 2048 blocks/windows)
                                <1>       ;    ESI = user's buffer address
                                <1>       ;        contains 64 bit block data
                                <1>       ;        BLOCK ADDRESS - (row, col), dword
                                <1>       ;        (first 32 bits)
                                <1>       ;        BLOCK SIZE - (rows, cols), dword
                                <1>       ;        (second 32 bits)
                                <1>       ; OUTPUT:
                                <1>       ;     [u.r0] will be > 0 if succesful
                                <1>
                                <1>       ; Window option ([v_ops] bit 4) will be ignored
                                <1>       ; (Function is used for display page coordinates)
                                <1>
0000F0D7 F605[88120300]20       <1>       test  byte [v_ops], 20h ; masked or direct ?
0000F0DE 755A                   <1>       jnz   short pix_op_blk_u ; blocks from user's buffer
                                <1>
0000F0E0 89F0                   <1>       mov   eax, esi ; destination position (row, col)
0000F0E2 E8DEF6FFFF             <1>       call  calc_pixel_offset
0000F0E7 3B05[8E120300]         <1>       cmp   eax, [v_siz]
0000F0ED 734A                   <1>       jnb   short pix_op_blk_retn ; out of display page
0000F0EF 89C6                   <1>       mov   esi, eax
0000F0F1 E8ACF6FFFF             <1>       call  pixels_to_byte_count
0000F0F6 89C7                   <1>       mov   edi, eax
0000F0F8 89D0                   <1>       mov   eax, edx  ; size
0000F0FA E8C6F6FFFF             <1>       call  calc_pixel_offset
                                <1>       ; 22/02/2021
0000F0FF 3B05[8E120300]         <1>       cmp   eax, [v_siz]
0000F105 7732                   <1>       ja    short pix_op_blk_retn ; out of display page
0000F107 01C6                   <1>       add   esi, eax
0000F109 3B35[8E120300]         <1>       cmp   esi, [v_siz]
0000F10F 7728                   <1>       ja    short pix_op_blk_retn ; out of display page
                                <1>
0000F111 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>
0000F117 E8FCF5FFFF             <1>       call  sysvideo_15_12 ; window preparations
0000F11C 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>
0000F11E 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>
0000F124 29CA                   <1>       sub   edx, ecx ; screen width - window width
                                <1> pix_op_blk_0:
0000F126 89C8                   <1>       mov   eax, ecx
0000F128 0105[64030300]         <1>       add   [u.r0], eax
0000F12E F3A4                   <1>       rep   movsb
0000F130 89C1                   <1>       mov   ecx, eax
0000F132 01D6                   <1>       add   esi, edx ; next row
0000F134 01D7                   <1>       add   edi, edx ; next row
0000F136 4B                     <1>       dec   ebx
0000F137 75ED                   <1>       jnz   short pix_op_blk_0
                                <1> pix_op_blk_retn:
0000F139 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
0000F13A 81FA00080000           <1>       cmp   edx, 2048
0000F140 7605                   <1>       jna   short pix_op_blk_u_0
                                <1>       ; Maximum 2048 blocks
0000F142 BA00080000             <1>       mov   edx, 2048
                                <1> pix_op_blk_u_0:
0000F147 8025[88120300]DF       <1>       and   byte [v_ops], ~20h ; clear masked bit
0000F14E 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:
0000F154 52                     <1>       push  edx
0000F155 B908000000             <1>       mov   ecx, 8
0000F15A BF[A2120300]           <1>       mov   edi, buffer8 ; 8 bytes small buffer
                                <1>       ; esi = user's buffer address
0000F15F E82D2A0000             <1>       call  transfer_from_user_buffer
0000F164 72D3                   <1>       jc    short pix_op_blk_retn
0000F166 01CE                   <1>       add   esi, ecx ; 22/02/2021
0000F168 56                     <1>       push  esi
0000F169 8B15[A2120300]         <1>       mov   edx, [buffer8] ; block start pos (row,col)
0000F16F 8B35[A6120300]         <1>       mov   esi, [buffer8+4] ; block size (rows,cols)
0000F175 8B0D[9A120300]         <1>       mov   ecx, [maskcolor]
0000F17B E848FCFFFF             <1>       call  pix_op_new_w ; new (change) color (window)
0000F180 5E                     <1>       pop   esi
                                <1>       ;pop   ebp
```

```
                                <1>       ;dec   ebp
0000F181 5A                     <1>       pop    edx
0000F182 4A                     <1>       dec    edx
0000F183 75CF                   <1>       jnz    short pix_op_blk_u_next
0000F185 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
0000F186 F605[88120300]20       <1>       test   byte [v_ops], 20h ; masked or direct ?
0000F18D 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
0000F18F 81FA00080000           <1>       cmp    edx, 2048
0000F195 7605                   <1>       jna    short pix_op_lin_u_0
                                <1>       ; Maximum 2048 lines
0000F197 BA00080000             <1>       mov    edx, 2048
                                <1> pix_op_lin_u_0:
0000F19C 890D[9A120300]         <1>       mov    [maskcolor], ecx ; save pixel color
0000F1A2 89D5                   <1>       mov    ebp, edx ; save line count
                                <1> pix_op_lin_u_next:
0000F1A4 B908000000             <1>       mov    ecx, 8
0000F1A9 BF[A2120300]           <1>       mov    edi, buffer8 ; 8 bytes small buffer
                                <1>       ; esi = user's buffer address
0000F1AE E8DE290000             <1>       call   transfer_from_user_buffer
0000F1B3 721E                   <1>       jc     short pix_op_lin_retn
0000F1B5 01CE                   <1>       add    esi, ecx ; 11/02/2021
0000F1B7 56                     <1>       push   esi
0000F1B8 8B35[A2120300]         <1>       mov    esi, [buffer8] ; line start pos (row,col)
0000F1BE 8B15[A6120300]         <1>       mov    edx, [buffer8+4] ; line length
0000F1C4 8B0D[9A120300]         <1>       mov    ecx, [maskcolor]
0000F1CA E805000000             <1>       call   pix_op_lin_vh ; new (change) color (window)
0000F1CF 5E                     <1>       pop    esi
0000F1D0 4D                     <1>       dec    ebp
0000F1D1 75D1                   <1>       jnz    short pix_op_lin_u_next
                                <1> pix_op_lin_retn:
0000F1D3 C3                     <1>       retn
                                <1>
                                <1> pix_op_lin_vh:
0000F1D4 81FA38140000           <1>       cmp    edx, 1438h ; 1920*1080 (780hx438h) limit
0000F1DA 7761                   <1>       ja     short pix_op_lin_err1 ; invalid type
                                <1>                           ; (for current version)
0000F1DC 66F7C2FF0F             <1>       test   dx, 0FFFh
0000F1E1 745A                   <1>       jz     short pix_op_lin_err1 ; zero length!
                                <1>
0000F1E3 89F0                   <1>       mov    eax, esi ; start point (row, col)
0000F1E5 E8DBF5FFFF             <1>       call   calc_pixel_offset
0000F1EA 3B05[8E120300]         <1>       cmp    eax, [v_siz]
0000F1F0 734B                   <1>       jnb    short pix_op_lin_err1 ; out of display page!
0000F1F2 E8ABF5FFFF             <1>       call   pixels_to_byte_count
0000F1F7 89C7                   <1>       mov    edi, eax ; start point offset
0000F1F9 033D[8A120300]         <1>       add    edi, [v_mem] ; LFB start address
0000F1FF 89C8                   <1>       mov    eax, ecx ; color
                                <1>
0000F201 F6C610                 <1>       test   dh, 10h
0000F204 0F848A000000           <1>       jz     pix_op_lin_h ; Horizontal line
                                <1>
                                <1> pix_op_lin_v:
                                <1>       ; Vertical line
0000F20A 80E60F                 <1>       and    dh, 0Fh ; low 12 bits
0000F20D 51                     <1>       push   ecx ; color
0000F20E 89D1                   <1>       mov    ecx, edx
0000F210 0FB705[86120300]       <1>       movzx  eax, word [v_width]
0000F217 89C3                   <1>       mov    ebx, eax
                                <1>       ; 12/02/2021
0000F219 F7E2                   <1>       mul    edx ; rows * [v_width]
```

```
5916 0000F21B 01F8                <1>       add    eax, edi
5917 0000F21D 3B05[96120300]      <1>       cmp    eax, [v_end]
5918 0000F223 58                  <1>       pop    eax ; color
5919 0000F224 7717                <1>       ja     short pix_op_lin_err1 ; out of display page
5920                              <1>       ; ecx = rows
5921 0000F226 89CA                <1>       mov    edx, ecx
5922                              <1>
5923 0000F228 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
5924 0000F22F 770D                <1>       ja     short pix_op_lin_v_2
5925                              <1>       ; 256 colors (1 byte per pixel)
5926 0000F231 010D[64030300]      <1>       add    [u.r0], ecx ; byte count
5927                              <1> pix_op_lin_v_1:
5928 0000F237 8807                <1>       mov    [edi], al
5929 0000F239 01DF                <1>       add    edi, ebx  ; next row
5930 0000F23B E2FA                <1>       loop   pix_op_lin_v_1
5931                              <1> pix_op_lin_err1:
5932 0000F23D C3                  <1>       retn
5933                              <1>
5934                              <1> pix_op_lin_v_2:
5935 0000F23E 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
5936 0000F245 773A                <1>       ja     short pix_op_lin_v_6 ; 32bpp
5937 0000F247 7226                <1>       jb     short pix_op_lin_v_4 ; 16bpp
5938                              <1>
5939                              <1>       ; 24 bit true colors
5940                              <1>       ; * 3
5941 0000F249 53                  <1>       push   ebx ; screen width in pixels
5942 0000F24A D1E3                <1>       shl    ebx, 1
5943 0000F24C 011C24              <1>       add    [esp], ebx
5944 0000F24F 5B                  <1>       pop    ebx ; screen width in bytes
5945 0000F250 010D[64030300]      <1>       add    [u.r0], ecx
5946 0000F256 D1E2                <1>       shl    edx, 1
5947 0000F258 0115[64030300]      <1>       add    [u.r0], edx ; byte count
5948                              <1> pix_op_lin_v_3:
5949 0000F25E 668907              <1>       mov    [edi], ax
5950 0000F261 C1C810              <1>       ror    eax, 16
5951 0000F264 884702              <1>       mov    [edi+2], al
5952 0000F267 C1C010              <1>       rol    eax, 16
5953 0000F26A 01DF                <1>       add    edi, ebx  ; next row
5954 0000F26C E2F0                <1>       loop   pix_op_lin_v_3
5955 0000F26E C3                  <1>       retn
5956                              <1>
5957                              <1> pix_op_lin_v_4:
5958                              <1>       ; 16 bit (65536) colors
5959 0000F26F D1E3                <1>       shl    ebx, 1
5960 0000F271 D1E2                <1>       shl    edx, 1
5961 0000F273 0115[64030300]      <1>       add    [u.r0], edx
5962                              <1> pix_op_lin_v_5:
5963 0000F279 668907              <1>       mov    [edi], ax
5964 0000F27C 01DF                <1>       add    edi, ebx  ; next row
5965 0000F27E E2F9                <1>       loop   pix_op_lin_v_5
5966 0000F280 C3                  <1>       retn
5967                              <1>
5968                              <1> pix_op_lin_v_6:
5969                              <1>       ; 32 bit true colors
5970 0000F281 C1E302              <1>       shl    ebx, 2
5971 0000F284 C1E202              <1>       shl    edx, 2
5972 0000F287 0115[64030300]      <1>       add    [u.r0], edx ; byte count
5973                              <1> pix_op_lin_v_7:
5974 0000F28D 8907                <1>       mov    [edi], eax
5975 0000F28F 01DF                <1>       add    edi, ebx  ; next row
5976 0000F291 E2FA                <1>       loop   pix_op_lin_v_7
5977 0000F293 C3                  <1>       retn
5978                              <1>
5979                              <1> pix_op_lin_h:
5980                              <1>       ; Horizontal line
5981 0000F294 80E60F              <1>       and    dh, 0Fh ; low 12 bits
5982 0000F297 89D1                <1>       mov    ecx, edx
5983 0000F299 6601D6              <1>       add    si, dx ; start column + columns
5984 0000F29C 663B35[86120300]    <1>       cmp    si, [v_width] ; screen width
5985 0000F2A3 7711                <1>       ja     short pix_op_lin_err2 ; out of columns limit
5986                              <1>
5987 0000F2A5 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
5988 0000F2AC 7709                <1>       ja     short pix_op_lin_h_1
5989                              <1>       ; 256 colors (1 byte per pixel)
5990 0000F2AE 010D[64030300]      <1>       add    [u.r0], ecx
5991 0000F2B4 F3AA                <1>       rep    stosb
5992                              <1> pix_op_lin_err2:
5993 0000F2B6 C3                  <1>       retn
5994                              <1>
5995                              <1> pix_op_lin_h_1:
5996 0000F2B7 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
5997 0000F2BE 7728                <1>       ja     short pix_op_lin_h_4 ; 32bpp
5998 0000F2C0 721A                <1>       jb     short pix_op_lin_h_3 ; 16bpp
5999                              <1>
6000                              <1>       ; 24 bit true colors
6001                              <1>       ; * 3
6002 0000F2C2 0115[64030300]      <1>       add    [u.r0], edx
6003 0000F2C8 D1E2                <1>       shl    edx, 1
6004 0000F2CA 0115[64030300]      <1>       add    [u.r0], edx
6005                              <1> pix_op_lin_h_2:
6006 0000F2D0 66AB                <1>       stosw
6007 0000F2D2 C1C810              <1>       ror    eax, 16
6008 0000F2D5 AA                  <1>       stosb
6009 0000F2D6 C1C010              <1>       rol    eax, 16
6010 0000F2D9 E2F5                <1>       loop   pix_op_lin_h_2
6011 0000F2DB C3                  <1>       retn
6012                              <1>
6013                              <1> pix_op_lin_h_3:
6014                              <1>       ; 16 bit (65536) colors
6015 0000F2DC D1E2                <1>       shl    edx, 1
6016 0000F2DE 0115[64030300]      <1>       add    [u.r0], edx
6017 0000F2E4 F366AB              <1>       rep    stosw
6018 0000F2E7 C3                  <1>       retn
6019                              <1>
6020                              <1> pix_op_lin_h_4:
```

```
                                <1>       ; 32 bit true colors
0000F2E8 C1E202                 <1>       shl    edx, 2
0000F2EB 0115[64030300]         <1>       add    [u.r0], edx
0000F2F1 F3AB                   <1>       rep    stosd
0000F2F3 C3                     <1>       retn
                                <1>
                                <1> pix_op_new_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; CHANGE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = color
                                <1>       ; edi = start pixel address
                                <1>
0000F2F4 F3AA                   <1>       rep    stosb
0000F2F6 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>
0000F2F7 F366AB                 <1>       rep    stosw
0000F2FA 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>
0000F2FB 66AB                   <1>       stosw
0000F2FD C1C810                 <1>       ror    eax, 16
0000F300 AA                     <1>       stosb
0000F301 C1C010                 <1>       rol    eax, 16
0000F304 E2F5                   <1>       loop   pix_op_new_24
0000F306 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>
0000F307 F3AB                   <1>       rep    stosd
0000F309 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>
0000F30A 88C4                   <1>       mov    ah, al
                                <1> pix_op_add_8_0:
0000F30C 0207                   <1>       add    al, [edi]
0000F30E 7302                   <1>       jnc    short pix_op_add_8_1
0000F310 B0FF                   <1>       mov    al, 0FFh ; Max. value
                                <1> pix_op_add_8_1:
0000F312 AA                     <1>       stosb
0000F313 88E0                   <1>       mov    al, ah
0000F315 E2F5                   <1>       loop   pix_op_add_8_0
0000F317 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>
0000F318 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_16_0:
0000F31A 660307                 <1>       add    ax, [edi]
0000F31D 7304                   <1>       jnc    short pix_op_add_16_1
0000F31F 66B8FFFF               <1>       mov    ax, 0FFFFh ; Max. value
                                <1> pix_op_add_16_1:
0000F323 66AB                   <1>       stosw
0000F325 89D0                   <1>       mov    eax, edx
0000F327 E2F1                   <1>       loop   pix_op_add_16_0
0000F329 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>
0000F32A 53                     <1>       push   ebx
0000F32B BBFFFFFF00             <1>       mov    ebx, 0FFFFFFh
                                <1>       ;and   eax, ebx ; 0FFFFFFh
0000F330 89C2                   <1>       mov    edx, eax
                                <1> pix_op_add_24_0:
0000F332 8B07                   <1>       mov    eax, [edi]
0000F334 21D8                   <1>       and    eax, ebx ; 0FFFFFFh
0000F336 01D0                   <1>       add    eax, edx
0000F338 39D8                   <1>       cmp    eax, ebx
0000F33A 7602                   <1>       jna    short pix_op_add_24_1
0000F33C 89D8                   <1>       mov    eax, ebx ; 0FFFFFFh  ; Max. value
                                <1> pix_op_add_24_1:
```

```
6126 0000F33E 66AB              <1>       stosw
6127 0000F340 C1E810            <1>       shr    eax, 16
6128 0000F343 AA                <1>       stosb
6129 0000F344 E2EC              <1>       loop   pix_op_add_24_0
6130 0000F346 89D0              <1>       mov    eax, edx
6131 0000F348 5B                <1>       pop    ebx
6132 0000F349 C3                <1>       retn
6133                            <1>
6134                            <1> pix_op_add_32:
6135                            <1>       ; 32 bit true colors
6136                            <1>       ; ADD PIXEL COLOR
6137                            <1>       ; ecx = pixel count per row
6138                            <1>       ; eax = color
6139                            <1>       ; edi = start pixel address
6140                            <1>
6141 0000F34A 89C2              <1>       mov    edx, eax
6142                            <1> pix_op_add_32_0:
6143 0000F34C 0307              <1>       add    eax, [edi]
6144 0000F34E 7303              <1>       jnc    short pix_op_add_32_1
6145                            <1>       ;mov   eax, 0FFFFFFFFh ; Max. value
6146 0000F350 29C0              <1>       sub    eax, eax
6147 0000F352 48                <1>       dec    eax
6148                            <1> pix_op_add_32_1:
6149 0000F353 AB                <1>       stosd
6150 0000F354 89D0              <1>       mov    eax, edx
6151 0000F356 E2F4              <1>       loop   pix_op_add_32_0
6152 0000F358 C3                <1>       retn
6153                            <1>
6154                            <1> pix_op_sub_8:
6155                            <1>       ; 8 bit colors (256 colors)
6156                            <1>       ; SUBTRACT PIXEL COLOR
6157                            <1>       ; ecx = pixel count per row
6158                            <1>       ;  al = color
6159                            <1>       ; edi = start pixel address
6160                            <1>
6161 0000F359 88C4              <1>       mov    ah, al
6162                            <1> pix_op_sub_8_0:
6163 0000F35B 8A07              <1>       mov    al, [edi]
6164 0000F35D 28E0              <1>       sub    al, ah
6165 0000F35F 7302              <1>       jnb    short pix_op_sub_8_1
6166 0000F361 30C0              <1>       xor    al, al ; 0 ; Min. value
6167                            <1> pix_op_sub_8_1:
6168 0000F363 AA                <1>       stosb
6169 0000F364 E2F5              <1>       loop   pix_op_sub_8_0
6170 0000F366 88E0              <1>       mov    al, ah
6171 0000F368 C3                <1>       retn
6172                            <1>
6173                            <1> pix_op_sub_16:
6174                            <1>       ; 16 bit colors (65536 colors)
6175                            <1>       ; SUBTRACT PIXEL COLOR
6176                            <1>       ; ecx = pixel count per row
6177                            <1>       ;  ax = color
6178                            <1>       ; edi = start pixel address
6179                            <1>
6180 0000F369 89C2              <1>       mov    edx, eax
6181                            <1> pix_op_sub_16_0:
6182 0000F36B 668B07            <1>       mov    ax, [edi]
6183 0000F36E 6629D0            <1>       sub    ax, dx
6184 0000F371 7302              <1>       jnb    short pix_op_sub_16_1
6185 0000F373 31C0              <1>       xor    eax, eax ; 0 ; Min. value
6186                            <1> pix_op_sub_16_1:
6187 0000F375 66AB              <1>       stosw
6188 0000F377 E2F2              <1>       loop   pix_op_sub_16_0
6189 0000F379 89D0              <1>       mov    eax, edx
6190 0000F37B C3                <1>       retn
6191                            <1>
6192                            <1> pix_op_sub_24:
6193                            <1>       ; 24 bit true colors
6194                            <1>       ; SUBTRACT PIXEL COLOR
6195                            <1>       ; ecx = pixel count per row
6196                            <1>       ; eax = color
6197                            <1>       ; edi = start pixel address
6198                            <1>
6199                            <1>       ;and   eax, 0FFFFFFh
6200 0000F37C 89C2              <1>       mov    edx, eax
6201                            <1> pix_op_sub_24_0:
6202 0000F37E 8B07              <1>       mov    eax, [edi]
6203                            <1>       ; 27/02/2021
6204 0000F380 25FFFFFF00        <1>       and    eax, 0FFFFFFh
6205 0000F385 29D0              <1>       sub    eax, edx
6206 0000F387 7302              <1>       jnb    short pix_op_sub_24_1
6207 0000F389 31C0              <1>       xor    eax, eax ; 0 ; Min. value
6208                            <1> pix_op_sub_24_1:
6209 0000F38B 66AB              <1>       stosw
6210 0000F38D C1E810            <1>       shr    eax, 16
6211 0000F390 AA                <1>       stosb
6212 0000F391 E2EB              <1>       loop   pix_op_sub_24_0
6213 0000F393 89D0              <1>       mov    eax, edx
6214 0000F395 C3                <1>       retn
6215                            <1>
6216                            <1> pix_op_sub_32:
6217                            <1>       ; 32 bit true colors
6218                            <1>       ; SUBTRACT PIXEL COLOR
6219                            <1>       ; ecx = pixel count per row
6220                            <1>       ; eax = color
6221                            <1>       ; edi = start pixel address
6222                            <1>
6223 0000F396 89C2              <1>       mov    edx, eax
6224                            <1> pix_op_sub_32_0:
6225 0000F398 8B07              <1>       mov    eax, [edi]
6226 0000F39A 29D0              <1>       sub    eax, edx
6227 0000F39C 7302              <1>       jnb    short pix_op_sub_32_1
6228 0000F39E 31C0              <1>       xor    eax, eax ; 0 ; Min. value
6229                            <1> pix_op_sub_32_1:
6230 0000F3A0 AB                <1>       stosd
```

```
6231 0000F3A1 E2F5              <1>       loop   pix_op_sub_32_0
6232 0000F3A3 89D0              <1>       mov    eax, edx
6233 0000F3A5 C3                <1>       retn
6234                            <1>
6235                            <1> pix_op_or_8:
6236                            <1>       ; 8 bit colors (256 colors)
6237                            <1>       ; OR PIXEL COLOR
6238                            <1>       ; ecx = pixel count per row
6239                            <1>       ;  al = color
6240                            <1>       ; edi = start pixel address
6241                            <1>
6242                            <1> pix_op_or_8_0:
6243 0000F3A6 0807              <1>       or     [edi], al
6244 0000F3A8 47                <1>       inc    edi
6245 0000F3A9 E2FB              <1>       loop   pix_op_or_8_0
6246 0000F3AB C3                <1>       retn
6247                            <1>
6248                            <1> pix_op_or_16:
6249                            <1>       ; 16 bit colors (65536 colors)
6250                            <1>       ; OR PIXEL COLOR
6251                            <1>       ; ecx = pixel count per row
6252                            <1>       ;  ax = color
6253                            <1>       ; edi = start pixel address
6254                            <1>
6255                            <1> pix_op_or_16_0:
6256 0000F3AC 660907            <1>       or     [edi], ax
6257 0000F3AF 47                <1>       inc    edi
6258 0000F3B0 47                <1>       inc    edi
6259 0000F3B1 E2F9              <1>       loop   pix_op_or_16_0
6260 0000F3B3 C3                <1>       retn
6261                            <1>
6262                            <1> pix_op_or_24:
6263                            <1>       ; 24 bit true colors
6264                            <1>       ; OR PIXEL COLOR
6265                            <1>       ; ecx = pixel count per row
6266                            <1>       ; eax = color
6267                            <1>       ; edi = start pixel address
6268                            <1>
6269 0000F3B4 89C2              <1>       mov    edx, eax
6270                            <1> pix_op_or_24_0:
6271 0000F3B6 0B07              <1>       or     eax, [edi]
6272 0000F3B8 66AB              <1>       stosw
6273 0000F3BA C1E810            <1>       shr    eax, 16
6274 0000F3BD AA                <1>       stosb
6275 0000F3BE 89D0              <1>       mov    eax, edx
6276 0000F3C0 E2F4              <1>       loop   pix_op_or_24_0
6277 0000F3C2 C3                <1>       retn
6278                            <1>
6279                            <1> pix_op_or_32:
6280                            <1>       ; 32 bit true colors
6281                            <1>       ; OR PIXEL COLOR
6282                            <1>       ; ecx = pixel count per row
6283                            <1>       ; eax = color
6284                            <1>       ; edi = start pixel address
6285                            <1>
6286                            <1>       ;mov   edx, eax
6287                            <1> pix_op_or_32_0:
6288                            <1>       ;or    eax, [edi]
6289                            <1>       ;stosd
6290                            <1>       ;mov   eax, edx
6291 0000F3C3 0907              <1>       or     [edi], eax
6292 0000F3C5 83C704            <1>       add    edi, 4
6293 0000F3C8 E2F9              <1>       loop   pix_op_or_32_0
6294 0000F3CA C3                <1>       retn
6295                            <1>
6296                            <1> pix_op_and_8:
6297                            <1>       ; 8 bit colors (256 colors)
6298                            <1>       ; AND PIXEL COLOR
6299                            <1>       ; ecx = pixel count per row
6300                            <1>       ;  al = color
6301                            <1>       ; edi = start pixel address
6302                            <1>
6303                            <1> pix_op_and_8_0:
6304 0000F3CB 2007              <1>       and    [edi], al
6305 0000F3CD 47                <1>       inc    edi
6306 0000F3CE E2FB              <1>       loop   pix_op_and_8_0
6307 0000F3D0 C3                <1>       retn
6308                            <1>
6309                            <1> pix_op_and_16:
6310                            <1>       ; 16 bit colors (65536 colors)
6311                            <1>       ; AND PIXEL COLOR
6312                            <1>       ; ecx = pixel count per row
6313                            <1>       ;  ax = color
6314                            <1>       ; edi = start pixel address
6315                            <1>
6316                            <1> pix_op_and_16_0:
6317 0000F3D1 662107            <1>       and    [edi], ax
6318 0000F3D4 47                <1>       inc    edi
6319 0000F3D5 47                <1>       inc    edi
6320 0000F3D6 E2F9              <1>       loop   pix_op_and_16_0
6321 0000F3D8 C3                <1>       retn
6322                            <1>
6323                            <1> pix_op_and_24:
6324                            <1>       ; 24 bit true colors
6325                            <1>       ; AND PIXEL COLOR
6326                            <1>       ; ecx = pixel count per row
6327                            <1>       ; eax = color
6328                            <1>       ; edi = start pixel address
6329                            <1>
6330 0000F3D9 89C2              <1>       mov    edx, eax
6331                            <1> pix_op_and_24_0:
6332 0000F3DB 2307              <1>       and    eax, [edi]
6333 0000F3DD 66AB              <1>       stosw
6334 0000F3DF C1E810            <1>       shr    eax, 16
6335 0000F3E2 AA                <1>       stosb
```

```
6336 0000F3E3 89D0              <1>       mov   eax, edx
6337 0000F3E5 E2F4              <1>       loop  pix_op_and_24_0
6338 0000F3E7 C3                <1>       retn
6339                            <1>
6340                            <1> pix_op_and_32:
6341                            <1>       ; 32 bit true colors
6342                            <1>       ; AND PIXEL COLOR
6343                            <1>       ; ecx = pixel count per row
6344                            <1>       ; eax = color
6345                            <1>       ; edi = start pixel address
6346                            <1>
6347                            <1>       ;mov   edx, eax
6348                            <1> pix_op_and_32_0:
6349                            <1>       ;and   eax, [edi]
6350                            <1>       ;stosd
6351                            <1>       ;mov   eax, edx
6352 0000F3E8 2107              <1>       and   [edi], eax
6353 0000F3EA 83C704            <1>       add   edi, 4
6354 0000F3ED E2F9              <1>       loop  pix_op_and_32_0
6355 0000F3EF C3                <1>       retn
6356                            <1>
6357                            <1> pix_op_xor_8:
6358                            <1>       ; 8 bit colors (256 colors)
6359                            <1>       ; XOR PIXEL COLOR
6360                            <1>       ; ecx = pixel count per row
6361                            <1>       ;  al = color
6362                            <1>       ; edi = start pixel address
6363                            <1>
6364                            <1> pix_op_xor_8_0:
6365 0000F3F0 3007              <1>       xor   [edi], al
6366 0000F3F2 47                <1>       inc   edi
6367 0000F3F3 E2FB              <1>       loop  pix_op_xor_8_0
6368 0000F3F5 C3                <1>       retn
6369                            <1>
6370                            <1> pix_op_xor_16:
6371                            <1>       ; 16 bit colors (65536 colors)
6372                            <1>       ; XOR PIXEL COLOR
6373                            <1>       ; ecx = pixel count per row
6374                            <1>       ;  ax = color
6375                            <1>       ; edi = start pixel address
6376                            <1>
6377                            <1> pix_op_xor_16_0:
6378 0000F3F6 663107            <1>       xor   [edi], ax
6379 0000F3F9 47                <1>       inc   edi
6380 0000F3FA 47                <1>       inc   edi
6381 0000F3FB E2F9              <1>       loop  pix_op_xor_16_0
6382 0000F3FD C3                <1>       retn
6383                            <1>
6384                            <1> pix_op_xor_24:
6385                            <1>       ; 24 bit true colors
6386                            <1>       ; XOR PIXEL COLOR
6387                            <1>       ; ecx = pixel count per row
6388                            <1>       ; eax = color
6389                            <1>       ; edi = start pixel address
6390                            <1>
6391 0000F3FE 89C2              <1>       mov   edx, eax
6392                            <1> pix_op_xor_24_0:
6393 0000F400 3307              <1>       xor   eax, [edi]
6394 0000F402 66AB              <1>       stosw
6395 0000F404 C1E810            <1>       shr   eax, 16
6396 0000F407 AA                <1>       stosb
6397 0000F408 89D0              <1>       mov   eax, edx
6398 0000F40A E2F4              <1>       loop  pix_op_xor_24_0
6399 0000F40C C3                <1>       retn
6400                            <1>
6401                            <1> pix_op_xor_32:
6402                            <1>       ; 32 bit true colors
6403                            <1>       ; XOR PIXEL COLOR
6404                            <1>       ; ecx = pixel count per row
6405                            <1>       ; eax = color
6406                            <1>       ; edi = start pixel address
6407                            <1>
6408                            <1>       ;mov   edx, eax
6409                            <1> pix_op_xor_32_0:
6410                            <1>       ;xor   eax, [edi]
6411                            <1>       ;stosd
6412                            <1>       ;mov   eax, edx
6413 0000F40D 3107              <1>       xor   [edi], eax
6414 0000F40F 83C704            <1>       add   edi, 4
6415 0000F412 E2F9              <1>       loop  pix_op_xor_32_0
6416 0000F414 C3                <1>       retn
6417                            <1>
6418                            <1> pix_op_mix_8:
6419                            <1>       ; 8 bit colors (256 colors)
6420                            <1>       ; MIX (AVERAGE) PIXEL COLORS
6421                            <1>       ; ecx = pixel count per row
6422                            <1>       ;  al = color
6423                            <1>       ; edi = start pixel address
6424                            <1>
6425 0000F415 88C4              <1>       mov   ah, al
6426                            <1> pix_op_mix_8_0:
6427 0000F417 0207              <1>       add   al, [edi]
6428 0000F419 D0D8              <1>       rcr   al, 1
6429 0000F41B AA                <1>       stosb
6430 0000F41C 88E0              <1>       mov   al, ah
6431 0000F41E E2F7              <1>       loop  pix_op_mix_8_0
6432 0000F420 C3                <1>       retn
6433                            <1>
6434                            <1> pix_op_mix_16:
6435                            <1>       ; 16 bit colors (65536 colors)
6436                            <1>       ; MIX (AVERAGE) PIXEL COLORS
6437                            <1>       ; ecx = pixel count per row
6438                            <1>       ;  ax = color
6439                            <1>       ; edi = start pixel address
6440                            <1>
```

```
6441 0000F421 89C2              <1>       mov    edx, eax
6442                            <1> pix_op_mix_16_0:
6443 0000F423 660307            <1>       add    ax, [edi]
6444 0000F426 66D1D8            <1>       rcr    ax, 1
6445 0000F429 66AB              <1>       stosw
6446 0000F42B 89D0              <1>       mov    eax, edx
6447 0000F42D E2F4              <1>       loop   pix_op_mix_16_0
6448 0000F42F C3                <1>       retn
6449                            <1>
6450                            <1> pix_op_mix_24:
6451                            <1>       ; 24 bit true colors
6452                            <1>       ; MIX (AVERAGE) PIXEL COLORS
6453                            <1>       ; ecx = pixel count per row
6454                            <1>       ; eax = color
6455                            <1>       ; edi = start pixel address
6456                            <1>
6457 0000F430 53                <1>       push   ebx
6458 0000F431 BBFFFFFF00        <1>       mov    ebx, 0FFFFFFh
6459                            <1>       ;and   eax, ebx ; 0FFFFFFh
6460 0000F436 89C2              <1>       mov    edx, eax
6461                            <1> pix_op_mix_24_0:
6462 0000F438 8B07              <1>       mov    eax, [edi]
6463 0000F43A 21D8              <1>       and    eax, ebx ; 0FFFFFFh
6464 0000F43C 01D0              <1>       add    eax, edx
6465 0000F43E D1E8              <1>       shr    eax, 1
6466                            <1>       ;rcr   eax, 1
6467 0000F440 66AB              <1>       stosw
6468 0000F442 C1E810            <1>       shr    eax, 16
6469 0000F445 AA                <1>       stosb
6470 0000F446 E2F0              <1>       loop   pix_op_mix_24_0
6471 0000F448 89D0              <1>       mov    eax, edx
6472 0000F44A 5B                <1>       pop    ebx
6473 0000F44B C3                <1>       retn
6474                            <1>
6475                            <1> pix_op_mix_32:
6476                            <1>       ; 32 bit true colors
6477                            <1>       ; MIX (AVERAGE) PIXEL COLORS
6478                            <1>       ; ecx = pixel count per row
6479                            <1>       ; eax = color
6480                            <1>       ; edi = start pixel address
6481                            <1>
6482 0000F44C 89C2              <1>       mov    edx, eax
6483                            <1> pix_op_mix_32_0:
6484 0000F44E 0307              <1>       add    eax, [edi]
6485 0000F450 D1D8              <1>       rcr    eax, 1
6486 0000F452 AB                <1>       stosd
6487 0000F453 89D0              <1>       mov    eax, edx
6488 0000F455 E2F7              <1>       loop   pix_op_mix_32_0
6489 0000F457 C3                <1>       retn
6490                            <1>
6491                            <1> pix_op_not_8:
6492                            <1>       ; 8 bit colors (256 colors)
6493                            <1>       ; NOT PIXEL COLOR
6494                            <1>       ; ecx = pixel count per row
6495                            <1>       ; edi = start pixel address
6496                            <1>
6497                            <1> pix_op_not_8_0:
6498 0000F458 F617              <1>       not    byte [edi]
6499 0000F45A 47                <1>       inc    edi
6500 0000F45B E2FB              <1>       loop   pix_op_not_8_0
6501 0000F45D C3                <1>       retn
6502                            <1>
6503                            <1> pix_op_not_16:
6504                            <1>       ; 16 bit colors (65536 colors)
6505                            <1>       ; NOT PIXEL COLOR
6506                            <1>       ; ecx = pixel count per row
6507                            <1>       ; edi = start pixel address
6508                            <1>
6509                            <1> pix_op_not_16_0:
6510 0000F45E 66F717            <1>       not    word [edi]
6511 0000F461 47                <1>       inc    edi
6512 0000F462 47                <1>       inc    edi
6513 0000F463 E2F9              <1>       loop   pix_op_not_16_0
6514 0000F465 C3                <1>       retn
6515                            <1>
6516                            <1> pix_op_not_24:
6517                            <1>       ; 24 bit true colors
6518                            <1>       ; NOT PIXEL COLOR
6519                            <1>       ; ecx = pixel count per row
6520                            <1>       ; edi = start pixel address
6521                            <1>
6522                            <1> pix_op_not_24_0:
6523 0000F466 66F717            <1>       not    word [edi]
6524 0000F469 47                <1>       inc    edi
6525 0000F46A 47                <1>       inc    edi
6526 0000F46B F617              <1>       not    byte [edi]
6527 0000F46D 47                <1>       inc    edi
6528 0000F46E E2F6              <1>       loop   pix_op_not_24_0
6529 0000F470 C3                <1>       retn
6530                            <1>
6531                            <1> pix_op_not_32:
6532                            <1>       ; 32 bit true colors
6533                            <1>       ; NOT PIXEL COLOR
6534                            <1>       ; ecx = pixel count per row
6535                            <1>       ; eax = color
6536                            <1>       ; edi = start pixel address
6537                            <1> pix_op_not_32_0:
6538 0000F471 F717              <1>       not    dword [edi]
6539 0000F473 83C704            <1>       add    edi, 4
6540 0000F476 E2F9              <1>       loop   pix_op_not_32_0
6541 0000F478 C3                <1>       retn
6542                            <1>
6543                            <1> pix_op_neg_8:
6544                            <1>       ; 8 bit colors (256 colors)
6545                            <1>       ; NEG PIXEL COLOR
```

```
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_neg_8_0:
0000F479 F61F                   <1>       neg   byte [edi]
0000F47B 47                     <1>       inc   edi
0000F47C E2FB                   <1>       loop  pix_op_neg_8_0
0000F47E 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:
0000F47F 66F71F                 <1>       neg   word [edi]
0000F482 47                     <1>       inc   edi
0000F483 47                     <1>       inc   edi
0000F484 E2F9                   <1>       loop  pix_op_neg_16_0
0000F486 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:
0000F487 8B07                   <1>       mov   eax, [edi]
0000F489 25FFFFFF00             <1>       and   eax, 0FFFFFFh
0000F48E F7D8                   <1>       neg   eax
0000F490 66AB                   <1>       stosw
0000F492 C1E810                 <1>       shr   eax, 16
0000F495 AA                     <1>       stosb
0000F496 E2EF                   <1>       loop  pix_op_neg_24_0
0000F498 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:
0000F499 F71F                   <1>       neg   dword [edi]
0000F49B 83C704                 <1>       add   edi, 4
0000F49E E2F9                   <1>       loop  pix_op_neg_32_0
0000F4A0 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:
0000F4A1 FE07                   <1>       inc   byte [edi]
0000F4A3 7502                   <1>       jnz   short pix_op_inc_8_1
                                <1>       ;mov  [edi], 0FFh ; Max. value
0000F4A5 FE0F                   <1>       dec   byte [edi]
                                <1> pix_op_inc_8_1:
0000F4A7 47                     <1>       inc   edi
0000F4A8 E2F7                   <1>       loop  pix_op_inc_8_0
0000F4AA 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:
0000F4AB 66FF07                 <1>       inc   word [edi]
0000F4AE 7503                   <1>       jnz   short pix_op_inc_16_1
                                <1>       ;mov  word [edi], 0FFFFh ; Max. value
0000F4B0 66FF0F                 <1>       dec   word [edi]
                                <1> pix_op_inc_16_1:
0000F4B3 47                     <1>       inc   edi
0000F4B4 47                     <1>       inc   edi
0000F4B5 E2F4                   <1>       loop  pix_op_inc_16_0
0000F4B7 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:
0000F4B8 8B07                   <1>       mov   eax, [edi]
0000F4BA 40                     <1>       inc   eax
0000F4BB 25FFFFFF00             <1>       and   eax, 0FFFFFFh
0000F4C0 7501                   <1>       jnz   short pix_op_inc_24_1
                                <1>       ;mov  eax, 0FFFFFFh  ; Max. value
0000F4C2 48                     <1>       dec   eax ; 0FFFFFFFFh
                                <1> pix_op_inc_24_1:
0000F4C3 66AB                   <1>       stosw
0000F4C5 C1E810                 <1>       shr   eax, 16
0000F4C8 AA                     <1>       stosb
0000F4C9 E2ED                   <1>       loop  pix_op_inc_24_0
0000F4CB 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:
0000F4CC FF07                   <1>       inc    dword [edi]
0000F4CE 7502                   <1>       jnz    short pix_op_inc_32_1
                                <1>       ;mov   dword [edi], 0FFFFFFFFh ; Max. value
0000F4D0 FF0F                   <1>       dec    dword [edi]
                                <1> pix_op_inc_32_1:
0000F4D2 83C704                 <1>       add    edi, 4
0000F4D5 E2F5                   <1>       loop   pix_op_inc_32_0
0000F4D7 C3                     <1>       retn
                                <1>
                                <1> pix_op_dec_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; DECREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_8_0:
0000F4D8 FE0F                   <1>       dec    byte [edi]
0000F4DA 7902                   <1>       jns    short pix_op_dec_8_1
0000F4DC FE07                   <1>       inc    byte [edi] ; 0 ; Min. value
                                <1> pix_op_dec_8_1:
0000F4DE 47                     <1>       inc    edi
0000F4DF E2F7                   <1>       loop   pix_op_dec_8_0
0000F4E1 C3                     <1>       retn
                                <1>
                                <1> pix_op_dec_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; DECREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_16_0:
0000F4E2 66FF0F                 <1>       dec    word [edi]
0000F4E5 7903                   <1>       jns    short pix_op_dec_16_1
0000F4E7 66FF07                 <1>       inc    word [edi] ; 0 ; Min. value
                                <1> pix_op_dec_16_1:
0000F4EA 47                     <1>       inc edi
0000F4EB 47                     <1>       inc edi
0000F4EC E2F4                   <1>       loop   pix_op_dec_16_0
0000F4EE C3                     <1>       retn
                                <1>
                                <1> pix_op_dec_24:
                                <1>       ; 24 bit true colors
                                <1>       ; DECREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_24_0:
0000F4EF 8B07                   <1>       mov    eax, [edi]
0000F4F1 25FFFFFF00             <1>       and    eax, 0FFFFFFh
0000F4F6 7401                   <1>       jz     short pix_op_dec_24_1
                                <1>                     ; 0 ; Min. value
0000F4F8 48                     <1>       dec    eax
                                <1> pix_op_dec_24_1:
0000F4F9 66AB                   <1>       stosw
0000F4FB C1E810                 <1>       shr    eax, 16
0000F4FE AA                     <1>       stosb
0000F4FF E2B7                   <1>       loop   pix_op_inc_24_0
0000F501 C3                     <1>       retn
                                <1>
                                <1> pix_op_dec_32:
                                <1>       ; 32 bit true colors
                                <1>       ; DECREASE PIXEL COLOR
                                <1>       ; ecx = pixel count per row
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_dec_32_0:
0000F502 FF0F                   <1>       dec    dword [edi]
0000F504 7902                   <1>       jns    short pix_op_dec_32_1
0000F506 FF07                   <1>       inc    dword [edi] ; 0 ; Min. value
                                <1> pix_op_dec_32_1:
0000F508 83C704                 <1>       add    edi, 4
0000F50B E2F5                   <1>       loop   pix_op_dec_32_0
0000F50D 89D0                   <1>       mov    eax, edx
0000F50F C3                     <1>       retn
                                <1>
                                <1> pix_op_rpl_8:
                                <1>       ; 8 bit colors (256 colors)
                                <1>       ; REPLACE PIXEL COLORS
                                <1>       ; ecx = pixel count per row
                                <1>       ;  al = new color
                                <1>       ; byte [maskcolor] = old color
                                <1>       ; edi = start pixel address
                                <1>
0000F510 8A25[9A120300]         <1>       mov    ah, [maskcolor]
                                <1> pix_op_rpl_8_0:
0000F516 3A27                   <1>       cmp    ah, [edi]
0000F518 7502                   <1>       jne    short pix_op_rpl_8_1
0000F51A 8807                   <1>       mov    [edi], al
                                <1> pix_op_rpl_8_1:
0000F51C 47                     <1>       inc    edi
0000F51D E2F7                   <1>       loop   pix_op_rpl_8_0
0000F51F C3                     <1>       retn
                                <1>
                                <1> pix_op_rpl_16:
                                <1>       ; 16 bit colors (65536 colors)
                                <1>       ; REPLACE PIXEL COLORS
                                <1>       ; ecx = pixel count per row
                                <1>       ;  ax = new color
                                <1>       ; word [maskcolor] = old color
                                <1>       ; edi = start pixel address
```

```
                                <1>
0000F520 8B15[9A120300]         <1>       mov   edx, [maskcolor]
                                <1> pix_op_rpl_16_0:
0000F526 663B17                 <1>       cmp   dx, [edi]
0000F529 7503                   <1>       jne   short pix_op_rpl_16_1
0000F52B 668907                 <1>       mov   [edi], ax
                                <1> pix_op_rpl_16_1:
0000F52E 47                     <1>       inc   edi
0000F52F 47                     <1>       inc   edi
0000F530 E2F4                   <1>       loop  pix_op_rpl_16_0
0000F532 C3                     <1>       retn
                                <1>
                                <1> pix_op_rpl_24:
                                <1>       ; 24 bit true colors
                                <1>       ; REPLACE PIXEL COLORS
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = new color
                                <1>       ; [maskcolor] = old color
                                <1>       ; edi = start pixel address
                                <1>
                                <1> pix_op_rpl_24_0:
0000F533 8B17                   <1>       mov   edx, [edi]
0000F535 81E2FFFFFF00           <1>       and   edx, 0FFFFFFh
0000F53B 3B15[9A120300]         <1>       cmp   edx, [maskcolor]
0000F541 7406                   <1>       je    short pix_op_rpl_24_1
0000F543 83C703                 <1>       add   edi, 3
0000F546 E2EB                   <1>       loop  pix_op_rpl_24_0
0000F548 C3                     <1>       retn
                                <1> pix_op_rpl_24_1:
0000F549 AA                     <1>       stosb
0000F54A C1C808                 <1>       ror   eax, 8
0000F54D 66AB                   <1>       stosw
0000F54F C1C008                 <1>       rol   eax, 8
0000F552 E2DF                   <1>       loop  pix_op_rpl_24_0
0000F554 C3                     <1>       retn
                                <1>
                                <1> pix_op_rpl_32:
                                <1>       ; 32 bit true colors
                                <1>       ; REPLACE PIXEL COLORS
                                <1>       ; ecx = pixel count per row
                                <1>       ; eax = new color
                                <1>       ; [maskcolor] = old color
                                <1>       ; edi = start pixel address
                                <1>
0000F555 8B15[9A120300]         <1>       mov   edx, [maskcolor]
                                <1> pix_op_rpl_32_0:
0000F55B 3B17                   <1>       cmp   edx, [edi]
0000F55D 7504                   <1>       jne   short pix_op_rpl_32_2
0000F55F AB                     <1>       stosd
0000F560 E2F9                   <1>       loop  pix_op_rpl_32_0
0000F562 C3                     <1>       retn
                                <1> pix_op_rpl_32_2:
0000F563 83C704                 <1>       add   edi, 4
0000F566 E2F3                   <1>       loop  pix_op_rpl_32_0
0000F568 C3                     <1>       retn
                                <1>
                                <1> pix_op_chr:
                                <1>       ; 15/02/2021
                                <1>       ; 05/02/2021
                                <1>       ; WRITE CHARACTER (FONT)
                                <1>       ; 05/01/2021 ([ufont])
                                <1>       ; 01/01/2021
                                <1>       ;     CL = char's color (8 bit, 256 colors)
                                <1>       ;    ECX = char's color (16 bit and true colors)
                                <1>       ;     DL = Character's ASCII code
                                <1>       ;     DH bit 0 -> font height
                                <1>       ;        0 -> 8x16 character font
                                <1>       ;        1 -> 8x8 character font
                                <1>       ;     DH bit 1 & 2 -> scale
                                <1>       ;        0 = 1/1 (8 pixels per char row)
                                <1>       ;        1 = 2/1 (16 pixels per char row)
                                <1>       ;        2 = 3/1 (24 pixels per char row)
                                <1>       ;        3 = 4/1 (32 pixels per char row)
                                <1>       ;     DH bit 6 -> [ufont] option (1 = use [ufont])
                                <1>       ;     If DH bit 7 = 1
                                <1>       ;      USER FONT (from user buffer)
                                <1>       ;         DL = 0 -> 8x8 (width: 1 byte per row)
                                <1>       ;         DL = 1 -> 8x16
                                <1>       ;         DL = 2 -> 16x16 (width: 2 bytes)
                                <1>       ;         DL = 3 -> 16x32
                                <1>       ;         DL = 4 -> 24x24 (width: 3 bytes)
                                <1>       ;         DL = 5 -> 24x48
                                <1>       ;         DL = 6 -> 32x32 (width: 4 bytes)
                                <1>       ;         DL = 7 -> 32x64
                                <1>       ;         DL > 7 -> invalid (unused)
                                <1>       ;      EDI = user's font buffer address
                                <1>       ;         (NOTE: byte order is as row0,row1,row2..)
                                <1>       ;     ESI = start position (row, column) (*)
                                <1>       ;          (HW = row, SI = column)
                                <1>
0000F569 89F0                   <1>       mov   eax, esi ; start position
0000F56B E855F2FFFF             <1>       call  calc_pixel_offset
0000F570 3B05[8E120300]         <1>       cmp   eax, [v_siz]
0000F576 736D                   <1>       jnb   short pix_op_chr_err ; out of display page!
0000F578 E825F2FFFF             <1>       call  pixels_to_byte_count
                                <1>       ; eax = font start offset
0000F57D 0305[8A120300]         <1>       add   eax, [v_mem] ; LFB start address
0000F583 A3[92120300]           <1>       mov   [v_str], eax ; font start address
                                <1>
0000F588 890D[9A120300]         <1>       mov   [maskcolor], ecx ; save char's color
                                <1>
0000F58E 8835[88120300]         <1>       mov   [v_ops], dh
                                <1>
0000F594 81E6FFFF0000           <1>       and   esi, 0FFFFh
0000F59A 8935[A2120300]         <1>       mov   [buffer8], esi ; start column
```

```
                                <1>
0000F5A0 31DB                   <1>       xor    ebx, ebx ; 0
0000F5A2 31C0                   <1>       xor    eax, eax ; 15/02/2021
                                <1>
0000F5A4 F6C680                 <1>       test   dh, 80h
0000F5A7 7577                   <1>       jnz    short pix_op_chr_u ; user font
                                <1>
0000F5A9 80E63F                 <1>       and    dh, 3Fh ; clear bit 6, [UFONT] option bit
0000F5AC 7409                   <1>       jz     short pix_op_chr_0
                                <1>
0000F5AE 80FE07                 <1>       cmp    dh, 7
0000F5B1 7732                   <1>       ja     short pix_op_chr_err
                                <1>                     ; invalid (undefined) option
0000F5B3 88F4                   <1>       mov    ah, dh
0000F5B5 D0EC                   <1>       shr    ah, 1
                                <1>       ; ah = 0 to 3, scale
                                <1>       ;jmp   short pix_op_chr_font_pixels
                                <1>
                                <1> pix_op_chr_font_pixels:
                                <1>       ; 05/02/2021
                                <1>       ; write scaled font to buffer
                                <1>
                                <1>       ; DL = ASCII code of character
                                <1>       ; AH = scale
                                <1>       ; EDI = buffer address (kernel)
                                <1>
                                <1> pix_op_chr_0:
0000F5B7 88D3                   <1>       mov    bl, dl ; 15/02/2021
0000F5B9 31C9                   <1>       xor    ecx, ecx
0000F5BB B610                   <1>       mov    dh, 16
0000F5BD F605[88120300]01       <1>       test   byte [v_ops], 1 ; 8x8 font ?
0000F5C4 7428                   <1>       jz     short pix_op_chr_2 ; 8x16 font
0000F5C6 B608                   <1>       mov    dh, 8
0000F5C8 C1E303                 <1>       shl    ebx, 3 ; * 8
0000F5CB F605[88120300]40       <1>       test   byte [v_ops], 40h ; [ufont] option
0000F5D2 7412                   <1>       jz     short pix_op_chr_1  ; no
                                <1>       ; test 8x8 user font is ready flag
0000F5D4 F605[7E120300]01       <1>       test   byte [ufont], 1
0000F5DB 7409                   <1>       jz     short pix_op_chr_1 ; no
0000F5DD 81C300500900           <1>       add    ebx, VGAFONT8USER
0000F5E3 EB2C                   <1>       jmp    short pix_op_chr_fpos_0
                                <1> pix_op_chr_err:
0000F5E5 C3                     <1>       retn
                                <1> pix_op_chr_1:
0000F5E6 81C3[A8600100]         <1>       add    ebx, vgafont8 ; system font (8x8)
0000F5EC EB23                   <1>       jmp    short pix_op_chr_fpos_0
                                <1> pix_op_chr_2:
0000F5EE C1E304                 <1>       shl    ebx, 4 ; * 16
0000F5F1 F605[88120300]40       <1>       test   byte [v_ops], 40h ; [ufont] option
0000F5F8 7411                   <1>       jz     short pix_op_chr_3  ; no
                                <1>       ; test 8x16 user font is ready flag
0000F5FA F605[7E120300]02       <1>       test   byte [ufont], 2
0000F601 7408                   <1>       jz     short pix_op_chr_3 ; no
0000F603 81C300400900           <1>       add    ebx, VGAFONT16USER
0000F609 EB06                   <1>       jmp    short pix_op_chr_fpos_0
                                <1> pix_op_chr_3:
0000F60B 81C3[A8760100]         <1>       add    ebx, vgafont16 ; system font (8x16)
                                <1> pix_op_chr_fpos_0:
0000F611 20E4                   <1>       and    ah, ah
0000F613 754B                   <1>       jnz    short pix_op_chr_fpos_1 ; scale > 1
                                <1>       ; no scale (scale = 1)
0000F615 89DE                   <1>       mov    esi, ebx ; 15/02/2021
0000F617 88F1                   <1>       mov    cl, dh ; rows/height (16 or 8)
0000F619 B608                   <1>       mov    dh, 8  ; columns/width
0000F61B E9CD000000             <1>       jmp    pix_op_chr_f2p
                                <1> pix_op_chr_u:
                                <1>       ; write user defined font
0000F620 80FE80                 <1>       cmp    dh, 80h
0000F623 75C0                   <1>       jne    short pix_op_chr_err
0000F625 80FA07                 <1>       cmp    dl, 7
0000F628 77BB                   <1>       ja     short pix_op_chr_err
                                <1>
                                <1>       ; 16/02/2021
0000F62A 89FE                   <1>       mov    esi, edi ; user's font buffer
                                <1>
                                <1>       ;xor   eax, eax
                                <1>       ; eax = 0 ; 15/02/2021
0000F62C 88D4                   <1>       mov    ah, dl
0000F62E D0EC                   <1>       shr    ah, 1
0000F630 FEC4                   <1>       inc    ah
                                <1>       ; ah =  1 to 4
0000F632 88E0                   <1>       mov    al, ah
0000F634 C0E003                 <1>       shl    al, 3 ; * 8
                                <1>       ; al = 8,16,24,32
0000F637 88C3                   <1>       mov    bl, al
0000F639 88C7                   <1>       mov    bh, al
0000F63B F6E4                   <1>       mul    ah
                                <1>       ; ax = 8,32,72,128 bytes
0000F63D F6C201                 <1>       test   dl, 1
0000F640 7405                   <1>       jz     short pix_op_chr_u_0
0000F642 66D1E0                 <1>       shl    ax, 1 ; *2
                                <1>       ; ax = 16,32,144,256 bytes
0000F645 D0E7                   <1>       shl    bh, 1
                                <1> pix_op_chr_u_0:
                                <1>       ; eax = byte count
0000F647 89C1                   <1>       mov    ecx, eax
0000F649 BF00760900             <1>       mov    edi, VBE3SAVERESTOREBLOCK
                                <1>       ; esi = user buffer
0000F64E E83E250000             <1>       call   transfer_from_user_buffer
0000F653 7290                   <1>       jc     short pix_op_chr_err
                                <1>
0000F655 88F9                   <1>       mov    cl, bh ; rows/height
0000F657 88DE                   <1>       mov    dh, bl ; columns (width)
0000F659 89FE                   <1>       mov    esi, edi ; VBE3SAVERESTOREBLOCK
0000F65B E98D000000             <1>       jmp    pix_op_chr_f2p
```

```
                                <1>
                                <1> pix_op_chr_fpos_1:
                                <1>       ; 18/02/2021
                                <1>       ; scale > 1
0000F660 88F5                   <1>       mov    ch, dh ; 16 or 8
0000F662 BF00760900             <1>       mov    edi, VBE3SAVERESTOREBLOCK
0000F667 89FE                   <1>       mov    esi, edi
0000F669 FECC                   <1>       dec    ah
0000F66B 7523                   <1>       jnz    short pix_op_chr_fpos_5 ; scale > 2
                                <1>       ; scale = 2
                                <1> pix_op_chr_fpos_2:
0000F66D B108                   <1>       mov    cl, 8
0000F66F 8A13                   <1>       mov    dl, [ebx]
                                <1> pix_op_chr_fpos_3:
0000F671 66C1E002               <1>       shl    ax, 2
0000F675 D0E2                   <1>       shl    dl, 1
0000F677 7302                   <1>       jnc    short pix_op_chr_fpos_4
0000F679 0C03                   <1>       or     al, 3
                                <1> pix_op_chr_fpos_4:
0000F67B FEC9                   <1>       dec    cl
0000F67D 75F2                   <1>       jnz    short pix_op_chr_fpos_3
0000F67F 66AB                   <1>       stosw
                                <1>       ; 18/02/2021
0000F681 66AB                   <1>       stosw
0000F683 43                     <1>       inc    ebx
0000F684 FECD                   <1>       dec    ch
0000F686 75E5                   <1>       jnz    short pix_op_chr_fpos_2
                                <1>       ; scale = 2
0000F688 88F1                   <1>       mov    cl, dh ; 16 or 8 (height/rows)
0000F68A D0E1                   <1>       shl    cl, 1  ; 32 or 16 rows
0000F68C B610                   <1>       mov    dh, 16 ; columns (width)
0000F68E EB5D                   <1>       jmp    short pix_op_chr_f2p
                                <1> pix_op_chr_fpos_5:
0000F690 FECC                   <1>       dec    ah
0000F692 7538                   <1>       jnz    short pix_op_chr_fpos_9 ; scale = 4
                                <1>       ; scale = 3
                                <1> pix_op_chr_fpos_6:
0000F694 B108                   <1>       mov    cl, 8
0000F696 8A13                   <1>       mov    dl, [ebx]
                                <1> pix_op_chr_fpos_7:
0000F698 C1E003                 <1>       shl    eax, 3
0000F69B D0E2                   <1>       shl    dl, 1 ; 18/02/2021
0000F69D 7302                   <1>       jnc    short pix_op_chr_fpos_8
0000F69F 0C07                   <1>       or     al, 7
                                <1> pix_op_chr_fpos_8:
0000F6A1 FEC9                   <1>       dec    cl
0000F6A3 75F3                   <1>       jnz    short pix_op_chr_fpos_7
0000F6A5 66AB                   <1>       stosw
                                <1>       ; 18/02/2021
0000F6A7 C1C810                 <1>       ror    eax, 16
0000F6AA AA                     <1>       stosb
0000F6AB C1C010                 <1>       rol    eax, 16
0000F6AE 66AB                   <1>       stosw
0000F6B0 C1C810                 <1>       ror    eax, 16
0000F6B3 AA                     <1>       stosb
0000F6B4 C1C010                 <1>       rol    eax, 16
0000F6B7 66AB                   <1>       stosw
0000F6B9 C1E810                 <1>       shr    eax, 16 ; 27/02/2021
0000F6BC AA                     <1>       stosb
0000F6BD 43                     <1>       inc    ebx
                                <1>       ; 18/02/2021
0000F6BE FECD                   <1>       dec    ch
0000F6C0 75D2                   <1>       jnz    short pix_op_chr_fpos_6
                                <1>       ; scale = 3
0000F6C2 88F1                   <1>       mov    cl, dh ; 16 or 8 (height/rows)
0000F6C4 D0E1                   <1>       shl    cl, 1
0000F6C6 00F1                   <1>       add    cl, dh ; 48 or 24 rows
0000F6C8 B618                   <1>       mov    dh, 24 ; columns (width)
0000F6CA EB21                   <1>       jmp    short pix_op_chr_f2p
                                <1>
                                <1> pix_op_chr_fpos_9:
                                <1>       ; scale = 4
0000F6CC B108                   <1>       mov    cl, 8
0000F6CE 8A13                   <1>       mov    dl, [ebx]
                                <1> pix_op_chr_fpos_10:
                                <1>       ; 18/02/2021
0000F6D0 C1E004                 <1>       shl    eax, 4
0000F6D3 D0E2                   <1>       shl    dl, 1 ; 18/02/2021
0000F6D5 7302                   <1>       jnc    short pix_op_chr_fpos_11
0000F6D7 0C0F                   <1>       or     al, 0Fh ; or al, 15
                                <1> pix_op_chr_fpos_11:
0000F6D9 FEC9                   <1>       dec    cl
0000F6DB 75F3                   <1>       jnz    short pix_op_chr_fpos_10
0000F6DD AB                     <1>       stosd
                                <1>       ; 18/02/2021
0000F6DE AB                     <1>       stosd
0000F6DF AB                     <1>       stosd
0000F6E0 AB                     <1>       stosd
0000F6E1 43                     <1>       inc    ebx
0000F6E2 FECD                   <1>       dec    ch
0000F6E4 75E6                   <1>       jnz    short pix_op_chr_fpos_9
                                <1>       ; scale = 4
0000F6E6 88F1                   <1>       mov    cl, dh ; 16 or 8 (height/rows)
0000F6E8 C0E102                 <1>       shl    cl, 2 ; 64 or 32 rows
0000F6EB B620                   <1>       mov    dh, 32 ; columns (width)
                                <1>       ;jmp   short pix_op_chr_f2p
                                <1>
                                <1> pix_op_chr_f2p:
                                <1>       ; write font pixels
0000F6ED 8B3D[92120300]         <1>       mov    edi, [v_str]
                                <1>       ; 15/02/2021
                                <1> pix_op_chr_f2p_next:
0000F6F3 80FE08                 <1>       cmp    dh, 8
0000F6F6 7706                   <1>       ja     short pix_op_chr_f2p_24
                                <1> pix_op_chr_f2p_8:
```

```
7071 0000F6F8 AC                <1>       lodsb
7072 0000F6F9 C1E018            <1>       shl    eax, 24 ; 15/02/2021
7073 0000F6FC EB16              <1>       jmp    short pix_op_chr_f2p_0
7074                            <1> pix_op_chr_f2p_24:
7075 0000F6FE 80FE18            <1>       cmp    dh, 24
7076 0000F701 7710              <1>       ja     short pix_op_chr_f2p_32
7077 0000F703 7207              <1>       jb     short pix_op_chr_f2p_16
7078                            <1>       ; 27/02/2021
7079                            <1>       ;mov   eax, [esi]
7080 0000F705 AD                <1>       lodsd
7081 0000F706 C1E008            <1>       shl    eax, 8
7082                            <1>       ;add   esi, 3
7083 0000F709 4E                <1>       dec    esi
7084 0000F70A EB08              <1>       jmp    short pix_op_chr_f2p_0
7085                            <1> pix_op_chr_f2p_16:
7086 0000F70C 66AD              <1>       lodsw
7087 0000F70E C1E010            <1>       shl    eax, 16 ; 15/02/2021
7088 0000F711 EB01              <1>       jmp    short pix_op_chr_f2p_0
7089                            <1> pix_op_chr_f2p_32:
7090 0000F713 AD                <1>       lodsd
7091                            <1> pix_op_chr_f2p_0:
7092                            <1>       ; EAX = font row (8,16,24,32 pixels)
7093                            <1>       ;      (bits are shifted to left)
7094                            <1>       ; CL = rows
7095                            <1>       ; DH = bits per row (8,16,24,32)
7096 0000F714 8B1D[A2120300]    <1>       mov    ebx, [buffer8] ; start column
7097 0000F71A 57                <1>       push   edi ; *
7098 0000F71B 52                <1>       push   edx ; **
7099                            <1> pix_op_chr_f2p_1:
7100 0000F71C E82E000000        <1>       call   pix_op_chr_w_pixel
7101                            <1> pix_op_chr_f2p_2:
7102 0000F721 663B1D[86120300]  <1>       cmp    bx, [v_width] ; current column
7103 0000F728 7304              <1>       jnb    short pix_op_chr_f2p_3
7104 0000F72A FECE              <1>       dec    dh
7105 0000F72C 75EE              <1>       jnz    short pix_op_chr_f2p_1 ; next bit
7106                            <1> pix_op_chr_f2p_3:
7107                            <1>       ;mov   ebx, [buffer8]
7108 0000F72E 5A                <1>       pop    edx ; **
7109 0000F72F 58                <1>       pop    eax ; *
7110 0000F730 3B3D[96120300]    <1>       cmp    edi, [v_end]
7111 0000F736 7316              <1>       jnb    short pix_op_chr_f2p_4
7112 0000F738 FEC9              <1>       dec    cl
7113 0000F73A 7412              <1>       jz     short pix_op_chr_f2p_4
7114                            <1>       ; 27/02/2021
7115 0000F73C 89C7              <1>       mov    edi, eax
7116 0000F73E 0FB705[86120300]  <1>       movzx  eax, word [v_width]
7117 0000F745 E858F0FFFF        <1>       call   pixels_to_byte_count
7118 0000F74A 01C7              <1>       add    edi, eax ; next position
7119 0000F74C EBA5              <1>       jmp    short pix_op_chr_f2p_next
7120                            <1> pix_op_chr_f2p_4:
7121 0000F74E C3                <1>       retn
7122                            <1>
7123                            <1> pix_op_chr_w_pixel:
7124                            <1>       ; 15/02/2021
7125 0000F74F 89C5              <1>       mov    ebp, eax
7126 0000F751 A1[9A120300]      <1>       mov    eax, [maskcolor]
7127 0000F756 803D[89120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
7128 0000F75D 7711              <1>       ja     short pix_op_chr_wp_2
7129                            <1>       ; 256 colors (1 byte per pixel)
7130 0000F75F D1E5              <1>       shl    ebp, 1
7131 0000F761 7302              <1>       jnc    short pix_op_chr_wp_0
7132 0000F763 8807              <1>       mov    [edi], al
7133                            <1> pix_op_chr_wp_0:
7134 0000F765 47                <1>       inc    edi
7135 0000F766 FF05[64030300]    <1>       inc    dword [u.r0] ; +1
7136                            <1> pix_op_chr_wp_1:
7137 0000F76C 43                <1>       inc    ebx
7138 0000F76D 89E8              <1>       mov    eax, ebp
7139 0000F76F C3                <1>       retn
7140                            <1> pix_op_chr_wp_2:
7141 0000F770 803D[89120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
7142 0000F777 772E              <1>       ja     short pix_op_chr_wp_6 ; 32bpp
7143 0000F779 721C              <1>       jb     short pix_op_chr_wp_4 ; 16bpp
7144                            <1>       ; 24 bit true colors
7145                            <1>       ; * 3
7146 0000F77B D1E5              <1>       shl    ebp, 1
7147 0000F77D 7309              <1>       jnc    short pix_op_chr_wp_3
7148 0000F77F 668907            <1>       mov    [edi], ax
7149 0000F782 C1E810            <1>       shr    eax, 16
7150 0000F785 884702            <1>       mov    [edi+2], al
7151                            <1> pix_op_chr_wp_3:
7152 0000F788 B803000000        <1>       mov    eax, 3 ; 27/02/2021
7153 0000F78D 01C7              <1>       add    edi, eax  ; add edi, 3
7154 0000F78F 0105[64030300]    <1>       add    [u.r0], eax ; +3
7155                            <1>
7156 0000F795 EBD5              <1>       jmp    short pix_op_chr_wp_1
7157                            <1>
7158                            <1> pix_op_chr_wp_4:
7159                            <1>       ; 16 bit (65536) colors
7160 0000F797 D1E5              <1>       shl    ebp, 1
7161 0000F799 7303              <1>       jnc    short pix_op_chr_wp_5
7162 0000F79B 668907            <1>       mov    [edi], ax
7163                            <1> pix_op_chr_wp_5:
7164 0000F79E 47                <1>       inc    edi
7165 0000F79F FF05[64030300]    <1>       inc    dword [u.r0] ; +1
7166 0000F7A5 EBBE              <1>       jmp    short pix_op_chr_wp_0
7167                            <1>
7168                            <1> pix_op_chr_wp_6:
7169                            <1>       ; 32 bit true colors
7170 0000F7A7 D1E5              <1>       shl    ebp, 1
7171 0000F7A9 7302              <1>       jnc    short pix_op_chr_wp_7
7172 0000F7AB 8907              <1>       mov    [edi], eax
7173                            <1> pix_op_chr_wp_7:
7174 0000F7AD 31C0              <1>       xor    eax, eax
7175 0000F7AF B004              <1>       mov    al, 4
```

```
7176 0000F7B1 01C7                <1>       add    edi, eax ; add edi, 4
7177 0000F7B3 0105[64030300]      <1>       add    [u.r0], eax ; +4
7178 0000F7B9 EBB1                <1>       jmp    short pix_op_chr_wp_1
7179                              <1>
7180                              <1> m_pix_op_cpy:
7181                              <1>       ; 26/02/2021
7182                              <1>       ; 06/02/2021
7183                              <1>       ; MASKED COPY PIXELS (full screen)
7184                              <1>       ;
7185                              <1>       ; jump from pix_op_cpy
7186                              <1>       ;
7187                              <1>       ; INPUT:
7188                              <1>       ;   ecx = transfer count (bytes)
7189                              <1>       ;   edi = [v_mem] = start address of LFB
7190                              <1>       ;   esi = user's buffer address (virtual)
7191                              <1>       ;
7192                              <1>       ; OUTPUT:
7193                              <1>       ;   [u.r0] will be > 0 if succesful
7194                              <1>
7195                              <1>       ; Full screen masked copy
7196                              <1>
7197                              <1>       ; Modified regs: eax, ebx, edx, esi, edi, ecx
7198                              <1>
7199                              <1> m_pix_op_cpy_0:
7200                              <1>       ;push  ebx ; *** ; 26/02/2021
7201 0000F7BB 57                 <1>       push   edi ; **
7202 0000F7BC 51                 <1>       push   ecx ; *
7203 0000F7BD 81F9F8070000       <1>       cmp    ecx, 2040 ; (3*680) ; 26/02/2021
7204 0000F7C3 7605               <1>       jna    short m_pix_op_cpy_1
7205 0000F7C5 B9F8070000         <1>       mov    ecx, 2040
7206                              <1> m_pix_op_cpy_1:
7207 0000F7CA BF00760900         <1>       mov    edi, VBE3SAVERESTOREBLOCK ; temporary buff
7208 0000F7CF E8BD230000         <1>       call   transfer from user buffer
7209 0000F7D4 726C               <1>       jc     short m_pix_op_cpy_3
7210 0000F7D6 01CE               <1>       add    esi, ecx
7211 0000F7D8 89F5               <1>       mov    ebp, esi  ; save user's buffer address
7212 0000F7DA 89FE               <1>       mov    esi, edi
7213 0000F7DC 89CB               <1>       mov    ebx, ecx
7214 0000F7DE 59                 <1>       pop    ecx ; *
7215 0000F7DF 29D9               <1>       sub    ecx, ebx
7216 0000F7E1 5F                 <1>       pop    edi ; **
7217 0000F7E2 31D2               <1>       xor    edx, edx ; 26/02/2021
7218 0000F7E4 803D[89120300]08   <1>       cmp    byte [v_bpp], 8
7219 0000F7EB 7435               <1>       je     short m_pix_op_cpy_1_8
7220 0000F7ED 803D[89120300]18   <1>       cmp    byte [v_bpp], 24
7221 0000F7F4 776D               <1>       ja     short m_pix_op_cpy_1_32
7222 0000F7F6 724D               <1>       jb     short m_pix_op_cpy_1_16
7223                              <1> m_pix_op_cpy_1_24:
7224                              <1>       ; 24 bit masked copy
7225                              <1>       ;mov   edx, 3
7226 0000F7F8 B203               <1>       mov    dl, 3 ; 26/02/2021
7227                              <1> m_pix_op_cpy_1_24_0:
7228 0000F7FA 66AD               <1>       lodsw
7229 0000F7FC C1E010             <1>       shl    eax, 16
7230 0000F7FF AC                 <1>       lodsb
7231 0000F800 C1C010             <1>       rol    eax, 16
7232 0000F803 3B05[9A120300]     <1>       cmp    eax, [maskcolor]
7233 0000F809 740F               <1>       je     short m_pix_op_cpy_1_24_1 ; exclude
7234 0000F80B 668907             <1>       mov    [edi], ax
7235 0000F80E C1E810             <1>       shr    eax, 16
7236 0000F811 884702             <1>       mov    [edi+2], al
7237 0000F814 0115[64030300]     <1>       add    [u.r0], edx ; +3
7238                              <1> m_pix_op_cpy_1_24_1:
7239 0000F81A 01D7               <1>       add    edi, edx ; +3
7240 0000F81C 29D3               <1>       sub    ebx, edx ; sub ebx, 3
7241 0000F81E 77DA               <1>       ja     short m_pix_op_cpy_1_24_0
7242 0000F820 EB15               <1>       jmp    short m_pix_op_cpy_2
7243                              <1>
7244                              <1> m_pix_op_cpy_1_8:
7245                              <1>       ; 8 bit masked copy
7246 0000F822 AC                 <1>       lodsb
7247 0000F823 3A05[9A120300]     <1>       cmp    al, [maskcolor]
7248 0000F829 7408               <1>       je     short m_pix_op_cpy_1_8_1 ; exclude
7249 0000F82B 8807               <1>       mov    [edi], al
7250 0000F82D FF05[64030300]     <1>       inc    dword [u.r0] ; +1
7251                              <1> m_pix_op_cpy_1_8_1:
7252 0000F833 47                 <1>       inc    edi ; +1
7253 0000F834 4B                 <1>       dec    ebx
7254 0000F835 75EB               <1>       jnz    short m_pix_op_cpy_1_8
7255                              <1> m_pix_op_cpy_2:
7256 0000F837 89EE               <1>       mov    esi, ebp ; restore user's buffer addr
7257 0000F839 09C9               <1>       or     ecx, ecx
7258                              <1>       ; 26/02/2021
7259 0000F83B 7407               <1>       jz     short m_pix_of_cpy_4
7260 0000F83D E979FFFFFF         <1>       jmp    m_pix_op_cpy_0
7261                              <1> m_pix_op_cpy_3:
7262 0000F842 59                 <1>       pop    ecx ; *
7263 0000F843 5F                 <1>       pop    edi ; **
7264                              <1> m_pix_of_cpy_4:
7265                              <1>       ;pop   ebx ; *** ; 26/02/2021
7266 0000F844 C3                 <1>       retn
7267                              <1>
7268                              <1> m_pix_op_cpy_1_16:
7269                              <1>       ; 16 bit masked copy
7270                              <1>       ;mov   edx, 2
7271 0000F845 B202               <1>       mov    dl, 2 ; 26/02/2021
7272                              <1> m_pix_op_cpy_1_16_0:
7273 0000F847 66AD               <1>       lodsw
7274 0000F849 663B05[9A120300]   <1>       cmp    ax, [maskcolor]
7275 0000F850 7409               <1>       je     short m_pix_op_cpy_1_16_1 ; exclude
7276 0000F852 668907             <1>       mov    [edi], ax
7277 0000F855 0115[64030300]     <1>       add    [u.r0], edx ; +2
7278                              <1> m_pix_op_cpy_1_16_1:
7279 0000F85B 01D7               <1>       add    edi, edx ; +2
7280 0000F85D 29D3               <1>       sub    ebx, edx ; sub ebx, 2
```

```
7281 0000F85F 77E6                <1>       ja     short m_pix_op_cpy_1_16_0
7282 0000F861 EBD4                <1>       jmp    short m_pix_op_cpy_2
7283                              <1>
7284                              <1> m_pix_op_cpy_1_32:
7285                              <1>       ; 32 bit masked copy
7286                              <1>       ;mov   edx, 4
7287 0000F863 B204                <1>       mov    dl, 4 ; 26/02/2021
7288                              <1> m_pix_op_cpy_1_32_0:
7289 0000F865 AD                  <1>       lodsd
7290 0000F866 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
7291 0000F86C 7408                <1>       je     short m_pix_op_cpy_1_32_1 ; exclude
7292 0000F86E 8907                <1>       mov    [edi], eax
7293 0000F870 0115[64030300]      <1>       add    [u.r0], edx ; +4
7294                              <1> m_pix_op_cpy_1_32_1:
7295 0000F876 01D7                <1>       add    edi, edx ; +4
7296 0000F878 29D3                <1>       sub    ebx, edx ; sub ebx, 4
7297 0000F87A 77E9                <1>       ja     short m_pix_op_cpy_1_32_0
7298 0000F87C EBB9                <1>       jmp    short m_pix_op_cpy_2
7299                              <1>
7300                              <1> m_pix_op_cpy_w:
7301                              <1>       ; 26/02/2021
7302                              <1>       ; 06/02/2021
7303                              <1>       ; MASKED COPY PIXELS (window)
7304                              <1>       ;
7305                              <1>       ; jump from pix_op_cpy_w
7306                              <1>       ;
7307                              <1>       ; INPUT:
7308                              <1>       ;   ecx = bytes per row (to be applied)
7309                              <1>       ;   edx = screen width in bytes
7310                              <1>       ;   ebx = row count
7311                              <1>       ;   esi = user's buffer address
7312                              <1>       ;   [v_str] = window start address
7313                              <1>       ;
7314                              <1>       ; OUTPUT:
7315                              <1>       ;   [u.r0] will be > 0 if succesful
7316                              <1>
7317                              <1>       ; Window masked copy
7318                              <1>
7319                              <1> m_pix_op_cpy_w_0:
7320 0000F87E 8B3D[92120300]      <1>       mov    edi, [v_str]
7321                              <1> m_pix_op_cpy_w_1:
7322 0000F884 57                  <1>       push   edi
7323 0000F885 56                  <1>       push   esi
7324 0000F886 53                  <1>       push   ebx
7325 0000F887 52                  <1>       push   edx
7326 0000F888 51                  <1>       push   ecx
7327 0000F889 E82DFFFFFF          <1>       call   m_pix_op_cpy ; 26/02/2021
7328 0000F88E 59                  <1>       pop    ecx
7329 0000F88F 5A                  <1>       pop    edx
7330 0000F890 5B                  <1>       pop    ebx
7331 0000F891 5E                  <1>       pop    esi
7332 0000F892 5F                  <1>       pop    edi
7333 0000F893 7209                <1>       jc     short m_pix_op_cpy_w_2
7334 0000F895 4B                  <1>       dec    ebx
7335 0000F896 7406                <1>       jz     short m_pix_op_cpy_w_2 ; ok.
7336                              <1>       ; next row
7337 0000F898 01CE                <1>       add    esi, ecx ; next row in user's buffer
7338 0000F89A 01D7                <1>       add    edi, edx ; next row of window (system)
7339 0000F89C EBE6                <1>       jmp    short m_pix_op_cpy_w_1
7340                              <1> m_pix_op_cpy_w_2:
7341 0000F89E C3                  <1>       retn
7342                              <1>
7343                              <1> m_pix_op_new:
7344                              <1>       ; 06/02/2021
7345                              <1>       ; CHANGE COLOR (MASKED, full screen)
7346                              <1>       ;
7347                              <1>       ; jump from pix_op_new
7348                              <1>       ;
7349                              <1>       ; INPUT:
7350                              <1>       ;   eax = color (AL, AX, EAX)
7351                              <1>       ;   ecx = [v_siz] ; display page pixel count
7352                              <1>       ;   esi = edi = [v_mem] ; LFB start address
7353                              <1>       ;
7354                              <1>       ;   [maskcolor] = mask color (to be excluded)
7355                              <1>       ;
7356                              <1>       ; OUTPUT:
7357                              <1>       ;     [u.r0] will be > 0 if succesful
7358                              <1>
7359                              <1>       ; Full screen
7360                              <1> m_pix_op_new_0:
7361 0000F89F 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7362 0000F8A6 7717                <1>       ja     short m_pix_op_new_1
7363                              <1>       ; 256 colors (8bpp)
7364                              <1>       ;jmp   short m_pix_op_new_8
7365                              <1> m_pix_op_new_8:
7366                              <1>       ; 8 bit colors (256 colors)
7367 0000F8A8 88C2                <1>       mov    dl, al ; new color
7368                              <1> m_pix_op_new_8_0:
7369 0000F8AA AC                  <1>       lodsb
7370 0000F8AB 3A05[9A120300]      <1>       cmp    al, [maskcolor]
7371 0000F8B1 7408                <1>       je     short m_pix_op_new_8_1 ; exclude
7372 0000F8B3 8817                <1>       mov    [edi], dl
7373 0000F8B5 FF05[64030300]      <1>       inc    dword [u.r0]
7374                              <1> m_pix_op_new_8_1:
7375 0000F8BB 47                  <1>       inc    edi
7376 0000F8BC E2EC                <1>       loop   m_pix_op_new_8_0
7377 0000F8BE C3                  <1>       retn
7378                              <1> m_pix_op_new_1:
7379 0000F8BF 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7380 0000F8C6 774B                <1>       ja     short m_pix_op_new_3 ; 32bpp
7381 0000F8C8 722C                <1>       jb     short m_pix_op_new_2 ; 16bpp
7382                              <1>       ; 24 bit true colors
7383                              <1>       ;jmp   short m_pix_op_new_24
7384                              <1> m_pix_op_new_24:
7385                              <1>       ; 24 bit true colors
```

```
7386 0000F8CA 89C2              <1>       mov    edx, eax ; new color
7387                            <1> m_pix_op_new_24_0:
7388 0000F8CC 66AD              <1>       lodsw
7389 0000F8CE C1E010            <1>       shl    eax, 16
7390 0000F8D1 AC                <1>       lodsb
7391 0000F8D2 C1C010            <1>       rol    eax, 16
7392 0000F8D5 3B05[9A120300]    <1>       cmp    eax, [maskcolor]
7393 0000F8DB 7413              <1>       je     short m_pix_op_new_24_1 ; exclude
7394 0000F8DD 668917            <1>       mov    [edi], dx
7395 0000F8E0 C1CA10            <1>       ror    edx, 16
7396 0000F8E3 885702            <1>       mov    [edi+2], dl
7397 0000F8E6 C1C210            <1>       rol    edx, 16
7398 0000F8E9 8305[64030300]03  <1>       add    dword [u.r0], 3
7399                            <1> m_pix_op_new_24_1:
7400 0000F8F0 83C703            <1>       add    edi, 3
7401 0000F8F3 E2D7              <1>       loop   m_pix_op_new_24_0
7402 0000F8F5 C3                <1>       retn
7403                            <1>       ; 65536 colors (16bpp)
7404                            <1> m_pix_op_new_2:
7405                            <1>       ;jmp   short m_pix_op_new_16
7406                            <1> m_pix_op_new_16:
7407                            <1>       ; 16 bit colors (65536 colors)
7408 0000F8F6 89C2              <1>       mov    edx, eax ; new color
7409                            <1> m_pix_op_new_16_0:
7410 0000F8F8 66AD              <1>       lodsw
7411 0000F8FA 663B05[9A120300]  <1>       cmp    ax, [maskcolor]
7412 0000F901 740A              <1>       je     short m_pix_op_new_16_1 ; exclude
7413 0000F903 668917            <1>       mov    [edi], dx
7414 0000F906 8305[64030300]02  <1>       add    dword [u.r0], 2
7415                            <1> m_pix_op_new_16_1:
7416 0000F90D 83C702            <1>       add    edi, 2
7417 0000F910 E2E6              <1>       loop   m_pix_op_new_16_0
7418 0000F912 C3                <1>       retn
7419                            <1> m_pix_op_new_3:
7420                            <1>       ; 32 bit true colors
7421                            <1>       ;jmp   short m_pix_op_new_32
7422                            <1> m_pix_op_new_32:
7423                            <1>       ; 32 bit true colors
7424 0000F913 89C2              <1>       mov    edx, eax ; new color
7425                            <1> m_pix_op_new_32_0:
7426 0000F915 AD                <1>       lodsd
7427 0000F916 3B05[9A120300]    <1>       cmp    eax, [maskcolor]
7428 0000F91C 7409              <1>       je     short m_pix_op_new_32_1 ; exclude
7429 0000F91E 8917              <1>       mov    [edi], edx
7430 0000F920 8305[64030300]04  <1>       add    dword [u.r0], 4
7431                            <1> m_pix_op_new_32_1:
7432 0000F927 83C704            <1>       add    edi, 4
7433 0000F92A E2E9              <1>       loop   m_pix_op_new_32_0
7434 0000F92C C3                <1>       retn
7435                            <1>
7436                            <1> m_pix_op_new_w:
7437                            <1>       ; 06/02/2021
7438                            <1>       ; CHANGE COLOR (MASKED, window)
7439                            <1>       ;
7440                            <1>       ; jump from pix_op_new_w
7441                            <1>       ;
7442                            <1>       ; INPUT:
7443                            <1>       ;   ecx = bytes per row (to be applied)
7444                            <1>       ;   edx = screen width in bytes
7445                            <1>       ;   ebx = row count
7446                            <1>       ;   eax = color
7447                            <1>       ;
7448                            <1>       ;   [maskcolor] = mask color (to be excluded)
7449                            <1>       ;
7450                            <1>       ; OUTPUT:
7451                            <1>       ;      [u.r0] will be > 0 if succesful
7452                            <1>
7453                            <1>       ; Window
7454                            <1>       ;mov   edi, [v_str] ; LFB start address
7455                            <1>       ;mov   esi, edi
7456                            <1>
7457 0000F92D 803D[89120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
7458 0000F934 7707              <1>       ja     short m_pix_op_new_w_1
7459                            <1>
7460                            <1>       ; 256 colors (8bpp)
7461 0000F936 BD[A8F80000]      <1>       mov    ebp, m_pix_op_new_8
7462 0000F93B EB1E              <1>       jmp    short m_pix_op_new_w_x
7463                            <1>
7464                            <1> m_pix_op_new_w_1:
7465 0000F93D 803D[89120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
7466 0000F944 7710              <1>       ja     short m_pix_op_new_w_3 ; 32bpp
7467 0000F946 7207              <1>       jb     short m_pix_op_new_w_2 ; 16bpp
7468                            <1>
7469                            <1>       ; 24 bit true colors
7470 0000F948 BD[CAF80000]      <1>       mov    ebp, m_pix_op_new_24
7471 0000F94D EB0C              <1>       jmp    short m_pix_op_new_w_x
7472                            <1>
7473                            <1>       ; 65536 colors (16bpp)
7474                            <1> m_pix_op_new_w_2:
7475 0000F94F BD[F6F80000]      <1>       mov    ebp, m_pix_op_new_16
7476 0000F954 EB05              <1>       jmp    short m_pix_op_new_w_x
7477                            <1>
7478                            <1>       ; 32 bit true colors
7479                            <1> m_pix_op_new_w_3:
7480 0000F956 BD[13F90000]      <1>       mov    ebp, m_pix_op_new_32
7481                            <1>       ;jmp   short m_pix_op_new_w_x
7482                            <1>
7483                            <1> m_pix_op_new_w_x:
7484                            <1> m_pix_op_add_w_x:
7485                            <1> m_pix_op_sub_w_x:
7486                            <1> m_pix_op_mix_w_x:
7487                            <1> m_pix_op_and_w_x:
7488                            <1> m_pix_op_orc_w_x:
7489                            <1> m_pix_op_xor_w_x:
7490                            <1> m_pix_op_not_w_x:
```

```
                                <1> m_pix_op_neg_w_x:
                                <1> m_pix_op_inc_w_x:
                                <1> m_pix_op_dec_w_x:
                                <1>       ; 27/02/2021
                                <1>       ; 26/02/2021
                                <1>       ; 06/02/2021
                                <1>       ; ecx = bytes per row (to be applied)
                                <1>       ; edx = windows (screen) width in bytes
                                <1>       ; ebx = row count
                                <1>       ; eax = color
                                <1>       ; ebp = pixel operation subroutine address
                                <1>       ; edi = esi = window start address
                                <1>
0000F95B 8B3D[92120300]         <1>       mov   edi, [v_str] ; LFB start address
0000F961 89FE                   <1>       mov   esi, edi
                                <1> m_pix_op_w_x_next:
0000F963 52                     <1>       push  edx
0000F964 51                     <1>       push  ecx
0000F965 56                     <1>       push  esi
0000F966 57                     <1>       push  edi
0000F967 50                     <1>       push  eax ; 26/02/2021
0000F968 8B0D[9E120300]         <1>       mov   ecx, [pixcount] ; 27/02/2021
0000F96E FFD5                   <1>       call  ebp ; call masked pixel-row operation
0000F970 58                     <1>       pop   eax ; 26/02/2021
0000F971 5F                     <1>       pop   edi
0000F972 5E                     <1>       pop   esi
0000F973 59                     <1>       pop   ecx
0000F974 5A                     <1>       pop   edx
0000F975 01D6                   <1>       add   esi, edx ; next row
0000F977 01D7                   <1>       add   edi, edx ; next row
0000F979 4B                     <1>       dec   ebx
0000F97A 75E7                   <1>       jnz   short m_pix_op_w_x_next
0000F97C C3                     <1>       retn
                                <1>
                                <1> m_pix_op_add:
                                <1>       ; 06/02/2021
                                <1>       ; ADD COLOR (MASKED, full screen)
                                <1>       ;
                                <1>       ; jump from pix_op_add
                                <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_add_0:
0000F97D 803D[89120300]08       <1>       cmp   byte [v_bpp], 8 ; 8bpp
0000F984 771C                   <1>       ja    short m_pix_op_add_1
                                <1>       ; 256 colors (8bpp)
                                <1>       ;jmp  short m_pix_op_add_8
                                <1> m_pix_op_add_8:
                                <1>       ; 8 bit colors (256 colors)
0000F986 88C2                   <1>       mov   dl, al ; new color
                                <1> m_pix_op_add_8_0:
0000F988 AC                     <1>       lodsb
0000F989 3A05[9A120300]         <1>       cmp   al, [maskcolor]
0000F98F 740D                   <1>       je    short m_pix_op_add_8_1 ; exclude
0000F991 FF05[64030300]         <1>       inc   dword [u.r0] ; +1
0000F997 0017                   <1>       add   [edi], dl
0000F999 7303                   <1>       jnc   short m_pix_op_add_8_1
0000F99B C607FF                 <1>       mov   byte [edi], 0FFh
                                <1> m_pix_op_add_8_1:
0000F99E 47                     <1>       inc   edi
0000F99F E2E7                   <1>       loop  m_pix_op_add_8_0
0000F9A1 C3                     <1>       retn
                                <1> m_pix_op_add_1:
0000F9A2 803D[89120300]18       <1>       cmp   byte [v_bpp], 24 ; 24bpp
0000F9A9 775E                   <1>       ja    short m_pix_op_add_3 ; 32bpp
0000F9AB 7238                   <1>       jb    short m_pix_op_add_2 ; 16bpp
                                <1>       ; 24 bit true colors
                                <1>       ;jmp  short m_pix_op_add_24
                                <1> m_pix_op_add_24:
                                <1>       ; 24 bit true colors
0000F9AD 89C2                   <1>       mov   edx, eax ; new color
0000F9AF 81CA000000FF           <1>       or    edx, 0FF000000h
                                <1> m_pix_op_add_24_0:
0000F9B5 66AD                   <1>       lodsw
0000F9B7 C1E010                 <1>       shl   eax, 16
0000F9BA AC                     <1>       lodsb
0000F9BB C1C010                 <1>       rol   eax, 16
0000F9BE 3B05[9A120300]         <1>       cmp   eax, [maskcolor]
0000F9C4 7419                   <1>       je    short m_pix_op_add_24_2 ; exclude
0000F9C6 8305[64030300]03       <1>       add   dword [u.r0], 3 ; +3
0000F9CD 01D0                   <1>       add   eax, edx
0000F9CF 7305                   <1>       jnc   short m_pix_op_add_24_1
0000F9D1 B8FFFFFF00             <1>       mov   eax, 0FFFFFFh
                                <1> m_pix_op_add_24_1:
0000F9D6 668907                 <1>       mov   [edi], ax
0000F9D9 C1E810                 <1>       shr   eax, 16
0000F9DC 884702                 <1>       mov   [edi+2], al
                                <1> m_pix_op_add_24_2:
0000F9DF 83C703                 <1>       add   edi, 3 ; +3
0000F9E2 E2D1                   <1>       loop  m_pix_op_add_24_0
0000F9E4 C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_add_2:
                                <1>       ;jmp  short m_pix_op_add_16
                                <1> m_pix_op_add_16:
                                <1>       ; 16 bit colors (65536 colors)
```

```
7596 0000F9E5 89C2                <1>       mov    edx, eax ; new color
7597                              <1> m_pix_op_add_16_0:
7598 0000F9E7 66AD                <1>       lodsw
7599 0000F9E9 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
7600 0000F9F0 7411                <1>       je     short m_pix_op_add_16_1 ; exclude
7601 0000F9F2 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
7602 0000F9F9 660117              <1>       add    [edi], dx
7603 0000F9FC 7305                <1>       jnc    short m_pix_op_add_16_1
7604 0000F9FE 66C707FFFF          <1>       mov    word [edi], 0FFFFh
7605                              <1> m_pix_op_add_16_1:
7606 0000FA03 83C702              <1>       add    edi, 2 ; +2
7607 0000FA06 E2DF                <1>       loop   m_pix_op_add_16_0
7608 0000FA08 C3                  <1>       retn
7609                              <1> m_pix_op_add_3:
7610                              <1>       ; 32 bit true colors
7611                              <1>       ;jmp   short m_pix_op_add_32
7612                              <1> m_pix_op_add_32:
7613                              <1>       ; 32 bit true colors
7614 0000FA09 89C2                <1>       mov    edx, eax ; new color
7615                              <1> m_pix_op_add_32_0:
7616 0000FA0B AD                  <1>       lodsd
7617 0000FA0C 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
7618 0000FA12 7411                <1>       je     short m_pix_op_add_32_1 ; exclude
7619 0000FA14 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
7620 0000FA1B 0117                <1>       add    [edi], edx
7621 0000FA1D 7306                <1>       jnc    short m_pix_op_add_32_1
7622 0000FA1F C707FFFFFFFF        <1>       mov    dword [edi], 0FFFFFFFFh
7623                              <1> m_pix_op_add_32_1:
7624 0000FA25 83C704              <1>       add    edi, 4 ; +4
7625 0000FA28 E2E1                <1>       loop   m_pix_op_add_32_0
7626 0000FA2A C3                  <1>       retn
7627                              <1>
7628                              <1> m_pix_op_add_w:
7629                              <1>       ; 06/02/2021
7630                              <1>       ; ADD COLOR (MASKED, window)
7631                              <1>       ;
7632                              <1>       ; jump from pix_op_add_w
7633                              <1>       ;
7634                              <1>       ; INPUT:
7635                              <1>       ;   ecx = bytes per row (to be applied)
7636                              <1>       ;   edx = screen width in bytes
7637                              <1>       ;   ebx = row count
7638                              <1>       ;   eax = color
7639                              <1>       ;
7640                              <1>       ;   [maskcolor] = mask color (to be excluded)
7641                              <1>       ;
7642                              <1>       ; OUTPUT:
7643                              <1>       ;      [u.r0] will be > 0 if succesful
7644                              <1>
7645                              <1>       ; window
7646                              <1>       ;mov   edi, [v_str] ; LFB start address
7647                              <1>       ;mov   esi, edi
7648                              <1>
7649 0000FA2B 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7650 0000FA32 7707                <1>       ja     short m_pix_op_add_w_1
7651                              <1>
7652                              <1>       ; 256 colors (8bpp)
7653 0000FA34 BD[86F90000]        <1>       mov    ebp, m_pix_op_add_8
7654 0000FA39 EB1E                <1>       jmp    short m_pix_op_add_w_4
7655                              <1>
7656                              <1> m_pix_op_add_w_1:
7657 0000FA3B 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7658 0000FA42 7710                <1>       ja     short m_pix_op_add_w_3 ; 32bpp
7659 0000FA44 7207                <1>       jb     short m_pix_op_add_w_2 ; 16bpp
7660                              <1>
7661                              <1>       ; 24 bit true colors
7662 0000FA46 BD[ADF90000]        <1>       mov    ebp, m_pix_op_add_24
7663 0000FA4B EB0C                <1>       jmp    short m_pix_op_add_w_4
7664                              <1>
7665                              <1>       ; 65536 colors (16bpp)
7666                              <1> m_pix_op_add_w_2:
7667 0000FA4D BD[E5F90000]        <1>       mov    ebp, m_pix_op_add_16
7668 0000FA52 EB05                <1>       jmp    short m_pix_op_add_w_4
7669                              <1>
7670                              <1>       ; 32 bit true colors
7671                              <1> m_pix_op_add_w_3:
7672 0000FA54 BD[09FA0000]        <1>       mov    ebp, m_pix_op_add_32
7673                              <1> m_pix_op_add_w_4:
7674 0000FA59 E9FDFEFFFF          <1>       jmp    m_pix_op_add_w_x
7675                              <1>
7676                              <1> m_pix_op_sub:
7677                              <1>       ; 02/03/2021
7678                              <1>       ; 06/02/2021
7679                              <1>       ; SUBTRACT COLOR (MASKED, full screen)
7680                              <1>       ;
7681                              <1>       ; jump from pix_op_sub
7682                              <1>       ;
7683                              <1>       ; INPUT:
7684                              <1>       ;   eax = color (AL, AX, EAX)
7685                              <1>       ;   ecx = [v_siz] ; display page pixel count
7686                              <1>       ;   esi = edi = [v_mem] ; LFB start address
7687                              <1>       ;
7688                              <1>       ;   [maskcolor] = mask color (to be excluded)
7689                              <1>       ;
7690                              <1>       ; OUTPUT:
7691                              <1>       ;      [u.r0] will be > 0 if succesful
7692                              <1>
7693                              <1>       ; Full screen
7694                              <1> m_pix_op_sub_0:
7695 0000FA5E 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7696 0000FA65 771C                <1>       ja     short m_pix_op_sub_1
7697                              <1>       ; 256 colors (8bpp)
7698                              <1>       ;jmp   short m_pix_op_sub_8
7699                              <1> m_pix_op_sub_8:
7700                              <1>       ; 8 bit colors (256 colors)
```

```
7701 0000FA67 88C2              <1>       mov   dl, al ; new color
7702                            <1> m_pix_op_sub_8_0:
7703 0000FA69 AC                <1>       lodsb
7704 0000FA6A 3A05[9A120300]    <1>       cmp   al, [maskcolor]
7705 0000FA70 740D              <1>       je    short m_pix_op_sub_8_1 ; exclude
7706 0000FA72 FF05[64030300]    <1>       inc   dword [u.r0] ; +1
7707 0000FA78 2817              <1>       sub   [edi], dl
7708 0000FA7A 7303              <1>       jnb   short m_pix_op_sub_8_1
7709 0000FA7C C60700            <1>       mov   byte [edi], 0
7710                            <1> m_pix_op_sub_8_1:
7711 0000FA7F 47                <1>       inc   edi
7712 0000FA80 E2E7              <1>       loop  m_pix_op_sub_8_0
7713 0000FA82 C3                <1>       retn
7714                            <1> m_pix_op_sub_1:
7715 0000FA83 803D[89120300]18  <1>       cmp   byte [v_bpp], 24 ; 24bpp
7716 0000FA8A 7755              <1>       ja    short m_pix_op_sub_3 ; 32bpp
7717 0000FA8C 722F              <1>       jb    short m_pix_op_sub_2 ; 16bpp
7718                            <1>       ; 24 bit true colors
7719                            <1>       ;jmp  short m_pix_op_sub_24
7720                            <1> m_pix_op_sub_24:
7721                            <1>       ; 24 bit true colors
7722 0000FA8E 89C2              <1>       mov   edx, eax ; new color
7723                            <1>       ; 02/03/2021
7724                            <1> m_pix_op_sub_24_0:
7725 0000FA90 66AD              <1>       lodsw
7726 0000FA92 C1E010            <1>       shl   eax, 16
7727 0000FA95 AC                <1>       lodsb
7728 0000FA96 C1C010            <1>       rol   eax, 16
7729 0000FA99 3B05[9A120300]    <1>       cmp   eax, [maskcolor]
7730 0000FA9F 7416              <1>       je    short m_pix_op_sub_24_2 ; exclude
7731 0000FAA1 8305[64030300]03  <1>       add   dword [u.r0], 3 ; +3
7732 0000FAA8 29D0              <1>       sub   eax, edx
7733 0000FAAA 7302              <1>       jnb   short m_pix_op_sub_24_1
7734 0000FAAC 31C0              <1>       xor   eax, eax ; 0
7735                            <1> m_pix_op_sub_24_1:
7736 0000FAAE 668907            <1>       mov   [edi], ax
7737 0000FAB1 C1E810            <1>       shr   eax, 16
7738 0000FAB4 884702            <1>       mov   [edi+2], al
7739                            <1> m_pix_op_sub_24_2:
7740 0000FAB7 83C703            <1>       add   edi, 3 ; +3
7741 0000FABA E2D4              <1>       loop  m_pix_op_sub_24_0
7742 0000FABC C3                <1>       retn
7743                            <1>       ; 65536 colors (16bpp)
7744                            <1> m_pix_op_sub_2:
7745                            <1>       ;jmp  short m_pix_op_sub_16
7746                            <1> m_pix_op_sub_16:
7747                            <1>       ; 16 bit colors (65536 colors)
7748 0000FABD 89C2              <1>       mov   edx, eax ; new color
7749                            <1> m_pix_op_sub_16_0:
7750 0000FABF 66AD              <1>       lodsw
7751 0000FAC1 663B05[9A120300]  <1>       cmp   ax, [maskcolor]
7752 0000FAC8 7411              <1>       je    short m_pix_op_sub_16_1 ; exclude
7753 0000FACA 8305[64030300]02  <1>       add   dword [u.r0], 2 ; +2
7754 0000FAD1 662917            <1>       sub   [edi], dx
7755 0000FAD4 7305              <1>       jnb   short m_pix_op_sub_16_1
7756 0000FAD6 31C0              <1>       xor   eax, eax
7757 0000FAD8 668907            <1>       mov   [edi], ax ; 0
7758                            <1> m_pix_op_sub_16_1:
7759 0000FADB 83C702            <1>       add   edi, 2 ; +2
7760 0000FADE E2DF              <1>       loop  m_pix_op_sub_16_0
7761 0000FAE0 C3                <1>       retn
7762                            <1> m_pix_op_sub_3:
7763                            <1>       ; 32 bit true colors
7764                            <1>       ;jmp  short m_pix_op_sub_32
7765                            <1> m_pix_op_sub_32:
7766                            <1>       ; 32 bit true colors
7767 0000FAE1 89C2              <1>       mov   edx, eax ; new color
7768                            <1> m_pix_op_sub_32_0:
7769 0000FAE3 AD                <1>       lodsd
7770 0000FAE4 3B05[9A120300]    <1>       cmp   eax, [maskcolor]
7771 0000FAEA 740F              <1>       je    short m_pix_op_sub_32_1 ; exclude
7772 0000FAEC 8305[64030300]04  <1>       add   dword [u.r0], 4 ; +4
7773 0000FAF3 2917              <1>       sub   [edi], edx
7774 0000FAF5 7304              <1>       jnb   short m_pix_op_sub_32_1
7775 0000FAF7 31C0              <1>       xor   eax, eax
7776 0000FAF9 8907              <1>       mov   [edi], eax ; 0
7777                            <1> m_pix_op_sub_32_1:
7778 0000FAFB 83C704            <1>       add   edi, 4 ; +4
7779 0000FAFE E2E3              <1>       loop  m_pix_op_sub_32_0
7780 0000FB00 C3                <1>       retn
7781                            <1>
7782                            <1> m_pix_op_sub_w:
7783                            <1>       ; 06/02/2021
7784                            <1>       ; SUBTRACT COLOR (MASKED, window)
7785                            <1>       ;
7786                            <1>       ; jump from pix_op_sub_w
7787                            <1>       ;
7788                            <1>       ; INPUT:
7789                            <1>       ;   ecx = bytes per row (to be applied)
7790                            <1>       ;   edx = screen width in bytes
7791                            <1>       ;   ebx = row count
7792                            <1>       ;   eax = color
7793                            <1>       ;
7794                            <1>       ;   [maskcolor] = mask color (to be excluded)
7795                            <1>       ;
7796                            <1>       ; OUTPUT:
7797                            <1>       ;     [u.r0] will be > 0 if succesful
7798                            <1>
7799                            <1>       ; window
7800                            <1>       ;mov  edi, [v_str] ; LFB start address
7801                            <1>       ;mov  esi, edi
7802                            <1>
7803 0000FB01 803D[89120300]08  <1>       cmp   byte [v_bpp], 8 ; 8bpp
7804 0000FB08 7707              <1>       ja    short m_pix_op_sub_w_1
7805                            <1>
```

```
                                <1>       ; 256 colors (8bpp)
0000FB0A BD[67FA0000]           <1>       mov    ebp, m_pix_op_sub_8
0000FB0F EB1E                   <1>       jmp    short m_pix_op_sub_w_4
                                <1>
                                <1> m_pix_op_sub_w_1:
0000FB11 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FB18 7710                   <1>       ja     short m_pix_op_sub_w_3 ; 32bpp
0000FB1A 7207                   <1>       jb     short m_pix_op_sub_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000FB1C BD[8EFA0000]           <1>       mov    ebp, m_pix_op_sub_24
0000FB21 EB0C                   <1>       jmp    short m_pix_op_sub_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_sub_w_2:
0000FB23 BD[BDFA0000]           <1>       mov    ebp, m_pix_op_sub_16
0000FB28 EB05                   <1>       jmp    short m_pix_op_sub_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_sub_w_3:
0000FB2A BD[E1FA0000]           <1>       mov    ebp, m_pix_op_sub_32
                                <1> m_pix_op_sub_w_4:
0000FB2F 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:
0000FB34 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FB3B 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)
0000FB3D 88C2                   <1>       mov    dl, al ; new (mixing) color
                                <1> m_pix_op_mix_8_0:
0000FB3F AC                     <1>       lodsb
0000FB40 3A05[9A120300]         <1>       cmp    al, [maskcolor]
0000FB46 740C                   <1>       je     short m_pix_op_mix_8_1 ; exclude
0000FB48 00D0                   <1>       add    al, dl ; 25/02/2021
0000FB4A D0D8                   <1>       rcr    al, 1
0000FB4C 8807                   <1>       mov    [edi], al
0000FB4E FF05[64030300]         <1>       inc    dword [u.r0] ; +1
                                <1> m_pix_op_mix_8_1:
0000FB54 47                     <1>       inc    edi
0000FB55 E2E8                   <1>       loop   m_pix_op_mix_8_0
0000FB57 C3                     <1>       retn
                                <1> m_pix_op_mix_1:
0000FB58 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FB5F 7752                   <1>       ja     short m_pix_op_mix_3 ; 32bpp
0000FB61 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
0000FB63 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFFFh
                                <1> m_pix_op_mix_24_0:
0000FB65 66AD                   <1>       lodsw
0000FB67 C1E010                 <1>       shl    eax, 16
0000FB6A AC                     <1>       lodsb
0000FB6B C1C010                 <1>       rol    eax, 16
0000FB6E 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FB74 7414                   <1>       je     short m_pix_op_mix_24_1 ; exclude
0000FB76 01D0                   <1>       add    eax, edx
0000FB78 D1E8                   <1>       shr    eax, 1
0000FB7A 668907                 <1>       mov    [edi], ax
0000FB7D C1E810                 <1>       shr    eax, 16
0000FB80 884702                 <1>       mov    [edi+2], al
0000FB83 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_mix_24_1:
0000FB8A 83C703                 <1>       add    edi, 3 ; +3
0000FB8D E2D6                   <1>       loop   m_pix_op_mix_24_0
0000FB8F 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)
0000FB90 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFh
                                <1> m_pix_op_mix_16_0:
0000FB92 66AD                   <1>       lodsw
0000FB94 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
0000FB9B 7410                   <1>       je     short m_pix_op_mix_16_1 ; exclude
0000FB9D 6601D0                 <1>       add    ax, dx
0000FBA0 66D1D8                 <1>       rcr    ax, 1
0000FBA3 668907                 <1>       mov    [edi], ax
0000FBA6 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
                                <1> m_pix_op_mix_16_1:
```

```
7911 0000FBAD 83C702              <1>       add    edi, 2 ; +2
7912 0000FBB0 E2E0                <1>       loop   m_pix_op_mix_16_0
7913 0000FBB2 C3                  <1>       retn
7914                              <1> m_pix_op_mix_3:
7915                              <1>       ; 32 bit true colors
7916                              <1>       ;jmp   short m_pix_op_mix_32
7917                              <1> m_pix_op_mix_32:
7918                              <1>       ; 32 bit true colors
7919 0000FBB3 89C2                <1>       mov    edx, eax ; new color
7920                              <1> m_pix_op_mix_32_0:
7921 0000FBB5 AD                  <1>       lodsd
7922 0000FBB6 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
7923 0000FBBC 740D                <1>       je     short m_pix_op_mix_32_1 ; exclude
7924 0000FBBE 01D0                <1>       add    eax, edx
7925                              <1>       ; 02/03/2021
7926 0000FBC0 D1D8                <1>       rcr    eax, 1
7927 0000FBC2 8907                <1>       mov    [edi], eax
7928 0000FBC4 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
7929                              <1> m_pix_op_mix_32_1:
7930 0000FBCB 83C704              <1>       add    edi, 4 ; +4
7931 0000FBCE E2E5                <1>       loop   m_pix_op_mix_32_0
7932 0000FBD0 C3                  <1>       retn
7933                              <1>
7934                              <1> m_pix_op_mix_w:
7935                              <1>       ; 06/02/2021
7936                              <1>       ; MIX COLOR (MASKED, window)
7937                              <1>       ;
7938                              <1>       ; jump from pix_op_mix_w
7939                              <1>       ;
7940                              <1>       ; INPUT:
7941                              <1>       ;   ecx = bytes per row (to be applied)
7942                              <1>       ;   edx = screen width in bytes
7943                              <1>       ;   ebx = row count
7944                              <1>       ;   eax = color
7945                              <1>       ;
7946                              <1>       ;   [maskcolor] = mask color (to be excluded)
7947                              <1>       ;
7948                              <1>       ; OUTPUT:
7949                              <1>       ;      [u.r0] will be > 0 if succesful
7950                              <1>
7951                              <1>       ; window
7952                              <1>       ;mov   edi, [v_str] ; LFB start address
7953                              <1>       ;mov   esi, edi
7954                              <1>
7955 0000FBD1 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
7956 0000FBD8 7707                <1>       ja     short m_pix_op_mix_w_1
7957                              <1>
7958                              <1>       ; 256 colors (8bpp)
7959 0000FBDA BD[3DFB0000]        <1>       mov    ebp, m_pix_op_mix_8
7960 0000FBDF EB1E                <1>       jmp    short m_pix_op_mix_w_4
7961                              <1>
7962                              <1> m_pix_op_mix_w_1:
7963 0000FBE1 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
7964 0000FBE8 7710                <1>       ja     short m_pix_op_mix_w_3 ; 32bpp
7965 0000FBEA 7207                <1>       jb     short m_pix_op_mix_w_2 ; 16bpp
7966                              <1>
7967                              <1>       ; 24 bit true colors
7968 0000FBEC BD[63FB0000]        <1>       mov    ebp, m_pix_op_mix_24
7969 0000FBF1 EB0C                <1>       jmp    short m_pix_op_mix_w_4
7970                              <1>
7971                              <1>       ; 65536 colors (16bpp)
7972                              <1> m_pix_op_mix_w_2:
7973 0000FBF3 BD[90FB0000]        <1>       mov    ebp, m_pix_op_mix_16
7974 0000FBF8 EB05                <1>       jmp    short m_pix_op_mix_w_4
7975                              <1>
7976                              <1>       ; 32 bit true colors
7977                              <1> m_pix_op_mix_w_3:
7978 0000FBFA BD[B3FB0000]        <1>       mov    ebp, m_pix_op_mix_32
7979                              <1> m_pix_op_mix_w_4:
7980 0000FBFF E957FDFFFF          <1>       jmp    m_pix_op_mix_w_x
7981                              <1>
7982                              <1> m_pix_op_and:
7983                              <1>       ; 06/02/2021
7984                              <1>       ; AND COLOR (MASKED, full screen)
7985                              <1>       ;
7986                              <1>       ; jump from pix_op_and
7987                              <1>       ;
7988                              <1>       ; INPUT:
7989                              <1>       ;   eax = color (AL, AX, EAX)
7990                              <1>       ;   ecx = [v_siz] ; display page pixel count
7991                              <1>       ;   esi = edi = [v_mem] ; LFB start address
7992                              <1>       ;
7993                              <1>       ;   [maskcolor] = mask color (to be excluded)
7994                              <1>       ;
7995                              <1>       ; OUTPUT:
7996                              <1>       ;      [u.r0] will be > 0 if succesful
7997                              <1>
7998                              <1>       ; Full screen
7999                              <1> m_pix_op_and_0:
8000 0000FC04 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8001 0000FC0B 7717                <1>       ja     short m_pix_op_and_1
8002                              <1>       ; 256 colors (8bpp)
8003                              <1>       ;jmp   short m_pix_op_and_8
8004                              <1> m_pix_op_and_8:
8005                              <1>       ; 8 bit colors (256 colors)
8006 0000FC0D 88C2                <1>       mov    dl, al ; new color
8007                              <1> m_pix_op_and_8_0:
8008 0000FC0F AC                  <1>       lodsb
8009 0000FC10 3A05[9A120300]      <1>       cmp    al, [maskcolor]
8010 0000FC16 7408                <1>       je     short m_pix_op_and_8_1 ; exclude
8011 0000FC18 2017                <1>       and    [edi], dl
8012 0000FC1A FF05[64030300]      <1>       inc    dword [u.r0] ; +1
8013                              <1> m_pix_op_and_8_1:
8014 0000FC20 47                  <1>       inc    edi
8015 0000FC21 E2EC                <1>       loop   m_pix_op_and_8_0
```

```
8016 0000FC23 C3                  <1>       retn
8017                              <1> m_pix_op_and_1:
8018 0000FC24 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8019 0000FC2B 774A                <1>       ja     short m_pix_op_and_3 ; 32bpp
8020 0000FC2D 722B                <1>       jb     short m_pix_op_and_2 ; 16bpp
8021                              <1>       ; 24 bit true colors
8022                              <1>       ;jmp   short m_pix_op_and_24
8023                              <1> m_pix_op_and_24:
8024                              <1>       ; 24 bit true colors
8025 0000FC2F 89C2                <1>       mov    edx, eax ; new color
8026                              <1>       ;and   edx, 0FFFFFFh
8027                              <1> m_pix_op_and_24_0:
8028 0000FC31 66AD                <1>       lodsw
8029 0000FC33 C1E010              <1>       shl    eax, 16
8030 0000FC36 AC                  <1>       lodsb
8031 0000FC37 C1C010              <1>       rol    eax, 16
8032 0000FC3A 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
8033 0000FC40 7412                <1>       je     short m_pix_op_and_24_1 ; exclude
8034 0000FC42 21D0                <1>       and    eax, edx
8035 0000FC44 668907              <1>       mov    [edi], ax
8036 0000FC47 C1E810              <1>       shr    eax, 16
8037 0000FC4A 884702              <1>       mov    [edi+2], al
8038 0000FC4D 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
8039                              <1> m_pix_op_and_24_1:
8040 0000FC54 83C703              <1>       add    edi, 3 ; +3
8041 0000FC57 E2D8                <1>       loop   m_pix_op_and_24_0
8042 0000FC59 C3                  <1>       retn
8043                              <1>       ; 65536 colors (16bpp)
8044                              <1> m_pix_op_and_2:
8045                              <1>       ;jmp   short m_pix_op_and_16
8046                              <1> m_pix_op_and_16:
8047                              <1>       ; 16 bit colors (65536 colors)
8048 0000FC5A 89C2                <1>       mov    edx, eax ; new color
8049                              <1>       ;and   edx, 0FFFFh
8050                              <1> m_pix_op_and_16_0:
8051 0000FC5C 66AD                <1>       lodsw
8052 0000FC5E 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
8053 0000FC65 740A                <1>       je     short m_pix_op_and_16_1 ; exclude
8054 0000FC67 662117              <1>       and    [edi], dx
8055 0000FC6A 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
8056                              <1> m_pix_op_and_16_1:
8057 0000FC71 83C702              <1>       add    edi, 2 ; +2
8058 0000FC74 E2E6                <1>       loop   m_pix_op_and_16_0
8059 0000FC76 C3                  <1>       retn
8060                              <1> m_pix_op_and_3:
8061                              <1>       ; 32 bit true colors
8062                              <1>       ;jmp   short m_pix_op_and_32
8063                              <1> m_pix_op_and_32:
8064                              <1>       ; 32 bit true colors
8065 0000FC77 89C2                <1>       mov    edx, eax ; new color
8066                              <1> m_pix_op_and_32_0:
8067 0000FC79 AD                  <1>       lodsd
8068 0000FC7A 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
8069 0000FC80 7409                <1>       je     short m_pix_op_and_32_1 ; exclude
8070 0000FC82 2117                <1>       and    [edi], edx ; 25/02/2021
8071 0000FC84 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
8072                              <1> m_pix_op_and_32_1:
8073 0000FC8B 83C704              <1>       add    edi, 4 ; +4
8074 0000FC8E E2E9                <1>       loop   m_pix_op_and_32_0
8075 0000FC90 C3                  <1>       retn
8076                              <1>
8077                              <1> m_pix_op_and_w:
8078                              <1>       ; 06/02/2021
8079                              <1>       ; AND COLOR (MASKED, window)
8080                              <1>       ;
8081                              <1>       ; jump from pix_op_and_w
8082                              <1>       ;
8083                              <1>       ; INPUT:
8084                              <1>       ;   ecx = bytes per row (to be applied)
8085                              <1>       ;   edx = screen width in bytes
8086                              <1>       ;   ebx = row count
8087                              <1>       ;   eax = color
8088                              <1>       ;
8089                              <1>       ;   [maskcolor] = mask color (to be excluded)
8090                              <1>       ;
8091                              <1>       ; OUTPUT:
8092                              <1>       ;      [u.r0] will be > 0 if succesful
8093                              <1>
8094                              <1>       ; window
8095                              <1>       ;mov   edi, [v_str] ; LFB start address
8096                              <1>       ;mov   esi, edi
8097                              <1>
8098 0000FC91 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8099 0000FC98 7707                <1>       ja     short m_pix_op_and_w_1
8100                              <1>
8101                              <1>       ; 256 colors (8bpp)
8102 0000FC9A BD[0DFC0000]        <1>       mov    ebp, m_pix_op_and_8
8103 0000FC9F EB1E                <1>       jmp    short m_pix_op_and_w_4
8104                              <1>
8105                              <1> m_pix_op_and_w_1:
8106 0000FCA1 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8107 0000FCA8 7710                <1>       ja     short m_pix_op_and_w_3 ; 32bpp
8108 0000FCAA 7207                <1>       jb     short m_pix_op_and_w_2 ; 16bpp
8109                              <1>
8110                              <1>       ; 24 bit true colors
8111 0000FCAC BD[2FFC0000]        <1>       mov    ebp, m_pix_op_and_24
8112 0000FCB1 EB0C                <1>       jmp    short m_pix_op_and_w_4
8113                              <1>
8114                              <1>       ; 65536 colors (16bpp)
8115                              <1> m_pix_op_and_w_2:
8116 0000FCB3 BD[5AFC0000]        <1>       mov    ebp, m_pix_op_and_16
8117 0000FCB8 EB05                <1>       jmp    short m_pix_op_and_w_4
8118                              <1>
8119                              <1>       ; 32 bit true colors
8120                              <1> m_pix_op_and_w_3:
```

```
8121 0000FCBA BD[77FC0000]        <1>       mov    ebp, m_pix_op_and_32
8122                              <1> m_pix_op_and_w_4:
8123 0000FCBF E997FCFFFF          <1>       jmp    m_pix_op_and_w_x
8124                              <1>
8125                              <1> m_pix_op_or:
8126                              <1>       ; 06/02/2021
8127                              <1>       ; OR COLOR (MASKED, full screen)
8128                              <1>       ;
8129                              <1>       ; jump from pix_op_orc
8130                              <1>       ;
8131                              <1>       ; INPUT:
8132                              <1>       ;   eax = color (AL, AX, EAX)
8133                              <1>       ;   ecx = [v_siz] ; display page pixel count
8134                              <1>       ;   esi = edi = [v_mem] ; LFB start address
8135                              <1>       ;
8136                              <1>       ;   [maskcolor] = mask color (to be excluded)
8137                              <1>       ;
8138                              <1>       ; OUTPUT:
8139                              <1>       ;      [u.r0] will be > 0 if succesful
8140                              <1>
8141                              <1>       ; Full screen
8142                              <1> m_pix_op_or_0:
8143 0000FCC4 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8144 0000FCCB 7717                <1>       ja     short m_pix_op_or_1
8145                              <1>       ; 256 colors (8bpp)
8146                              <1>       ;jmp   short m_pix_op_or_8
8147                              <1> m_pix_op_or_8:
8148                              <1>       ; 8 bit colors (256 colors)
8149 0000FCCD 88C2                <1>       mov    dl, al ; new color
8150                              <1> m_pix_op_or_8_0:
8151 0000FCCF AC                  <1>       lodsb
8152 0000FCD0 3A05[9A120300]      <1>       cmp    al, [maskcolor]
8153 0000FCD6 7408                <1>       je     short m_pix_op_or_8_1 ; exclude
8154 0000FCD8 0817                <1>       or     [edi], dl
8155 0000FCDA FF05[64030300]      <1>       inc    dword [u.r0] ; +1
8156                              <1> m_pix_op_or_8_1:
8157 0000FCE0 47                  <1>       inc    edi
8158 0000FCE1 E2EC                <1>       loop   m_pix_op_or_8_0
8159 0000FCE3 C3                  <1>       retn
8160                              <1> m_pix_op_or_1:
8161 0000FCE4 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8162 0000FCEB 774A                <1>       ja     short m_pix_op_or_3 ; 32bpp
8163 0000FCED 722B                <1>       jb     short m_pix_op_or_2 ; 16bpp
8164                              <1>       ; 24 bit true colors
8165                              <1>       ;jmp   short m_pix_op_or_24
8166                              <1> m_pix_op_or_24:
8167                              <1>       ; 24 bit true colors
8168 0000FCEF 89C2                <1>       mov    edx, eax ; new color
8169                              <1>       ;and   edx, 0FFFFFFh
8170                              <1> m_pix_op_or_24_0:
8171 0000FCF1 66AD                <1>       lodsw
8172 0000FCF3 C1E010              <1>       shl    eax, 16
8173 0000FCF6 AC                  <1>       lodsb
8174 0000FCF7 C1C010              <1>       rol    eax, 16
8175 0000FCFA 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
8176 0000FD00 7412                <1>       je     short m_pix_op_or_24_1 ; exclude
8177 0000FD02 09D0                <1>       or     eax, edx
8178 0000FD04 668907              <1>       mov    [edi], ax
8179 0000FD07 C1E810              <1>       shr    eax, 16
8180 0000FD0A 884702              <1>       mov    [edi+2], al
8181 0000FD0D 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
8182                              <1> m_pix_op_or_24_1:
8183 0000FD14 83C703              <1>       add    edi, 3 ; +3
8184 0000FD17 E2D8                <1>       loop   m_pix_op_or_24_0
8185 0000FD19 C3                  <1>       retn
8186                              <1>       ; 65536 colors (16bpp)
8187                              <1> m_pix_op_or_2:
8188                              <1>       ;jmp   short m_pix_op_or_16
8189                              <1> m_pix_op_or_16:
8190                              <1>       ; 16 bit colors (65536 colors)
8191 0000FD1A 89C2                <1>       mov    edx, eax ; new color
8192                              <1>       ;and   edx, 0FFFFh
8193                              <1> m_pix_op_or_16_0:
8194 0000FD1C 66AD                <1>       lodsw
8195 0000FD1E 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
8196 0000FD25 740A                <1>       je     short m_pix_op_or_16_1 ; exclude
8197 0000FD27 660917              <1>       or     [edi], dx
8198 0000FD2A 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
8199                              <1> m_pix_op_or_16_1:
8200 0000FD31 83C702              <1>       add    edi, 2 ; +2
8201 0000FD34 E2E6                <1>       loop   m_pix_op_or_16_0
8202 0000FD36 C3                  <1>       retn
8203                              <1> m_pix_op_or_3:
8204                              <1>       ; 32 bit true colors
8205                              <1>       ;jmp   short m_pix_op_or_32
8206                              <1> m_pix_op_or_32:
8207                              <1>       ; 32 bit true colors
8208 0000FD37 89C2                <1>       mov    edx, eax ; new color
8209                              <1> m_pix_op_or_32_0:
8210 0000FD39 AD                  <1>       lodsd
8211 0000FD3A 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
8212 0000FD40 7409                <1>       je     short m_pix_op_or_32_1 ; exclude
8213 0000FD42 0917                <1>       or     [edi], edx ; 25/02/2021
8214 0000FD44 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
8215                              <1> m_pix_op_or_32_1:
8216 0000FD4B 83C704              <1>       add    edi, 4 ; +4
8217 0000FD4E E2E9                <1>       loop   m_pix_op_or_32_0
8218 0000FD50 C3                  <1>       retn
8219                              <1>
8220                              <1> m_pix_op_or_w:
8221                              <1>       ; 06/02/2021
8222                              <1>       ; MIX COLOR (MASKED, window)
8223                              <1>       ;
8224                              <1>       ; jump from pix_op_or_w
8225                              <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>
0000FD51 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FD58 7707                   <1>       ja     short m_pix_op_or_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0000FD5A BD[CDFC0000]           <1>       mov    ebp, m_pix_op_or_8
0000FD5F EB1E                   <1>       jmp    short m_pix_op_or_w_4
                                <1>
                                <1> m_pix_op_or_w_1:
0000FD61 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FD68 7710                   <1>       ja     short m_pix_op_or_w_3 ; 32bpp
0000FD6A 7207                   <1>       jb     short m_pix_op_or_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0000FD6C BD[EFFC0000]           <1>       mov    ebp, m_pix_op_or_24
0000FD71 EB0C                   <1>       jmp    short m_pix_op_or_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_or_w_2:
0000FD73 BD[1AFD0000]           <1>       mov    ebp, m_pix_op_or_16
0000FD78 EB05                   <1>       jmp    short m_pix_op_or_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_or_w_3:
0000FD7A BD[37FD0000]           <1>       mov    ebp, m_pix_op_or_32
                                <1> m_pix_op_or_w_4:
0000FD7F 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:
0000FD84 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FD8B 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)
0000FD8D 88C2                   <1>       mov    dl, al ; new color
                                <1> m_pix_op_xor_8_0:
0000FD8F AC                     <1>       lodsb
0000FD90 3A05[9A120300]         <1>       cmp    al, [maskcolor]
0000FD96 7408                   <1>       je     short m_pix_op_xor_8_1 ; exclude
0000FD98 3017                   <1>       xor    [edi], dl
0000FD9A FF05[64030300]         <1>       inc    dword [u.r0] ; +1
                                <1> m_pix_op_xor_8_1:
0000FDA0 47                     <1>       inc    edi
0000FDA1 E2EC                   <1>       loop   m_pix_op_xor_8_0
0000FDA3 C3                     <1>       retn
                                <1> m_pix_op_xor_1:
0000FDA4 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FDAB 774A                   <1>       ja     short m_pix_op_xor_3 ; 32bpp
0000FDAD 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
0000FDAF 89C2                   <1>       mov    edx, eax ; new color
                                <1>       ;and   edx, 0FFFFFFh
                                <1> m_pix_op_xor_24_0:
0000FDB1 66AD                   <1>       lodsw
0000FDB3 C1E010                 <1>       shl    eax, 16
0000FDB6 AC                     <1>       lodsb
0000FDB7 C1C010                 <1>       rol    eax, 16
0000FDBA 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FDC0 7412                   <1>       je     short m_pix_op_xor_24_1 ; exclude
0000FDC2 31D0                   <1>       xor    eax, edx
0000FDC4 668907                 <1>       mov    [edi], ax
0000FDC7 C1E810                 <1>       shr    eax, 16
0000FDCA 884702                 <1>       mov    [edi+2], al
0000FDCD 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_xor_24_1:
0000FDD4 83C703                 <1>       add    edi, 3 ; +3
0000FDD7 E2D8                   <1>       loop   m_pix_op_xor_24_0
0000FDD9 C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_xor_2:
```

```
8331                                    <1>       ;jmp   short m_pix_op_xor_16
8332                                    <1> m_pix_op_xor_16:
8333                                    <1>       ; 16 bit colors (65536 colors)
8334 0000FDDA 89C2                     <1>       mov    edx, eax ; new color
8335                                    <1>       ;and   edx, 0FFFFh
8336                                    <1> m_pix_op_xor_16_0:
8337 0000FDDC 66AD                     <1>       lodsw
8338 0000FDDE 663B05[9A120300]         <1>       cmp    ax, [maskcolor]
8339 0000FDE5 740A                     <1>       je     short m_pix_op_xor_16_1 ; exclude
8340 0000FDE7 663117                   <1>       xor    [edi], dx
8341 0000FDEA 8305[64030300]02         <1>       add    dword [u.r0], 2 ; +2
8342                                    <1> m_pix_op_xor_16_1:
8343 0000FDF1 83C702                   <1>       add    edi, 2 ; +2
8344 0000FDF4 E2E6                     <1>       loop   m_pix_op_xor_16_0
8345 0000FDF6 C3                       <1>       retn
8346                                    <1> m_pix_op_xor_3:
8347                                    <1>       ; 32 bit true colors
8348                                    <1>       ;jmp   short m_pix_op_xor_32
8349                                    <1> m_pix_op_xor_32:
8350                                    <1>       ; 32 bit true colors
8351 0000FDF7 89C2                     <1>       mov    edx, eax ; new color
8352                                    <1> m_pix_op_xor_32_0:
8353 0000FDF9 AD                       <1>       lodsd
8354 0000FDFA 3B05[9A120300]           <1>       cmp    eax, [maskcolor]
8355 0000FE00 7409                     <1>       je     short m_pix_op_xor_32_1 ; exclude
8356 0000FE02 3117                     <1>       xor    [edi], edx ; 25/02/2021
8357 0000FE04 8305[64030300]04         <1>       add    dword [u.r0], 4 ; +4
8358                                    <1> m_pix_op_xor_32_1:
8359 0000FE0B 83C704                   <1>       add    edi, 4 ; +4
8360 0000FE0E E2E9                     <1>       loop   m_pix_op_xor_32_0
8361 0000FE10 C3                       <1>       retn
8362                                    <1>
8363                                    <1> m_pix_op_xor_w:
8364                                    <1>       ; 06/02/2021
8365                                    <1>       ; XOR COLOR (MASKED, window)
8366                                    <1>       ;
8367                                    <1>       ; jump from pix_op_xor_w
8368                                    <1>       ;
8369                                    <1>       ; INPUT:
8370                                    <1>       ;   ecx = bytes per row (to be applied)
8371                                    <1>       ;   edx = screen width in bytes
8372                                    <1>       ;   ebx = row count
8373                                    <1>       ;   eax = color
8374                                    <1>       ;
8375                                    <1>       ;   [maskcolor] = mask color (to be excluded)
8376                                    <1>       ;
8377                                    <1>       ; OUTPUT:
8378                                    <1>       ;      [u.r0] will be > 0 if succesful
8379                                    <1>
8380                                    <1>       ; window
8381                                    <1>       ;mov   edi, [v_str] ; LFB start address
8382                                    <1>       ;mov   esi, edi
8383                                    <1>
8384 0000FE11 803D[89120300]08         <1>       cmp    byte [v_bpp], 8 ; 8bpp
8385 0000FE18 7707                     <1>       ja     short m_pix_op_xor_w_1
8386                                    <1>
8387                                    <1>       ; 256 colors (8bpp)
8388 0000FE1A BD[8DFD0000]             <1>       mov    ebp, m_pix_op_xor_8
8389 0000FE1F EB1E                     <1>       jmp    short m_pix_op_xor_w_4
8390                                    <1>
8391                                    <1> m_pix_op_xor_w_1:
8392 0000FE21 803D[89120300]18         <1>       cmp    byte [v_bpp], 24 ; 24bpp
8393 0000FE28 7710                     <1>       ja     short m_pix_op_xor_w_3 ; 32bpp
8394 0000FE2A 7207                     <1>       jb     short m_pix_op_xor_w_2 ; 16bpp
8395                                    <1>
8396                                    <1>       ; 24 bit true colors
8397 0000FE2C BD[AFFD0000]             <1>       mov    ebp, m_pix_op_xor_24
8398 0000FE31 EB0C                     <1>       jmp    short m_pix_op_xor_w_4
8399                                    <1>
8400                                    <1>       ; 65536 colors (16bpp)
8401                                    <1> m_pix_op_xor_w_2:
8402 0000FE33 BD[DAFD0000]             <1>       mov    ebp, m_pix_op_xor_16
8403 0000FE38 EB05                     <1>       jmp    short m_pix_op_xor_w_4
8404                                    <1>
8405                                    <1>       ; 32 bit true colors
8406                                    <1> m_pix_op_xor_w_3:
8407 0000FE3A BD[F7FD0000]             <1>       mov    ebp, m_pix_op_xor_32
8408                                    <1> m_pix_op_xor_w_4:
8409 0000FE3F E917FBFFFF               <1>       jmp    m_pix_op_xor_w_x
8410                                    <1>
8411                                    <1> m_pix_op_not:
8412                                    <1>       ; 06/02/2021
8413                                    <1>       ; NOT COLOR (MASKED, full screen)
8414                                    <1>       ;
8415                                    <1>       ; jump from pix_op_not
8416                                    <1>       ;
8417                                    <1>       ; INPUT:
8418                                    <1>       ;   ecx = [v_siz] ; display page pixel count
8419                                    <1>       ;   esi = edi = [v_mem] ; LFB start address
8420                                    <1>       ;
8421                                    <1>       ;   [maskcolor] = mask color (to be excluded)
8422                                    <1>       ;
8423                                    <1>       ; OUTPUT:
8424                                    <1>       ;      [u.r0] will be > 0 if succesful
8425                                    <1>
8426                                    <1>       ; Full screen
8427                                    <1> m_pix_op_not_0:
8428 0000FE44 803D[89120300]08         <1>       cmp    byte [v_bpp], 8 ; 8bpp
8429 0000FE4B 7715                     <1>       ja     short m_pix_op_not_1
8430                                    <1>       ; 256 colors (8bpp)
8431                                    <1>       ;jmp   short m_pix_op_not_8
8432                                    <1> m_pix_op_not_8:
8433                                    <1>       ; 8 bit colors (256 colors)
8434 0000FE4D AC                       <1>       lodsb
8435 0000FE4E 3A05[9A120300]           <1>       cmp    al, [maskcolor]
```

```
8436 0000FE54 7408                <1>       je    short m_pix_op_not_8_1 ; exclude
8437 0000FE56 F617                <1>       not   byte [edi]
8438 0000FE58 FF05[64030300]      <1>       inc   dword [u.r0] ; +1
8439                              <1> m_pix_op_not_8_1:
8440 0000FE5E 47                  <1>       inc   edi
8441 0000FE5F E2EC                <1>       loop  m_pix_op_not_8
8442 0000FE61 C3                  <1>       retn
8443                              <1> m_pix_op_not_1:
8444 0000FE62 803D[89120300]18    <1>       cmp   byte [v_bpp], 24 ; 24bpp
8445 0000FE69 7746                <1>       ja    short m_pix_op_not_3 ; 32bpp
8446 0000FE6B 7229                <1>       jb    short m_pix_op_not_2 ; 16bpp
8447                              <1>       ; 24 bit true colors
8448                              <1>       ;jmp  short m_pix_op_not_24
8449                              <1> m_pix_op_not_24:
8450                              <1>       ; 24 bit true colors
8451 0000FE6D 66AD                <1>       lodsw
8452 0000FE6F C1E010              <1>       shl   eax, 16
8453 0000FE72 AC                  <1>       lodsb
8454 0000FE73 C1C010              <1>       rol   eax, 16
8455 0000FE76 3B05[9A120300]      <1>       cmp   eax, [maskcolor]
8456 0000FE7C 7412                <1>       je    short m_pix_op_not_24_1 ; exclude
8457 0000FE7E F7D0                <1>       not   eax
8458 0000FE80 668907              <1>       mov   [edi], ax
8459 0000FE83 C1E810              <1>       shr   eax, 16
8460 0000FE86 884702              <1>       mov   [edi+2], al
8461 0000FE89 8305[64030300]03    <1>       add   dword [u.r0], 3 ; +3
8462                              <1> m_pix_op_not_24_1:
8463 0000FE90 83C703              <1>       add   edi, 3 ; +3
8464 0000FE93 E2D8                <1>       loop  m_pix_op_not_24
8465 0000FE95 C3                  <1>       retn
8466                              <1>       ; 65536 colors (16bpp)
8467                              <1> m_pix_op_not_2:
8468                              <1>       ;jmp  short m_pix_op_not_16
8469                              <1> m_pix_op_not_16:
8470                              <1>       ; 16 bit colors (65536 colors)
8471 0000FE96 66AD                <1>       lodsw
8472 0000FE98 663B05[9A120300]    <1>       cmp   ax, [maskcolor]
8473 0000FE9F 740A                <1>       je    short m_pix_op_not_16_1 ; exclude
8474 0000FEA1 66F717              <1>       not   word [edi]
8475 0000FEA4 8305[64030300]02    <1>       add   dword [u.r0], 2 ; +2
8476                              <1> m_pix_op_not_16_1:
8477 0000FEAB 83C702              <1>       add   edi, 2 ; +2
8478 0000FEAE E2E6                <1>       loop  m_pix_op_not_16
8479 0000FEB0 C3                  <1>       retn
8480                              <1> m_pix_op_not_3:
8481                              <1>       ; 32 bit true colors
8482                              <1>       ;jmp  short m_pix_op_not_32
8483                              <1> m_pix_op_not_32:
8484                              <1>       ; 32 bit true colors
8485 0000FEB1 AD                  <1>       lodsd
8486 0000FEB2 3B05[9A120300]      <1>       cmp   eax, [maskcolor]
8487 0000FEB8 7409                <1>       je    short m_pix_op_not_32_1 ; exclude
8488 0000FEBA F717                <1>       not   dword [edi]
8489 0000FEBC 8305[64030300]04    <1>       add   dword [u.r0], 4 ; +4
8490                              <1> m_pix_op_not_32_1:
8491 0000FEC3 83C704              <1>       add   edi, 4 ; +4
8492 0000FEC6 E2E9                <1>       loop  m_pix_op_not_32
8493 0000FEC8 C3                  <1>       retn
8494                              <1>
8495                              <1> m_pix_op_not_w:
8496                              <1>       ; 06/02/2021
8497                              <1>       ; NOT COLOR (MASKED, window)
8498                              <1>       ;
8499                              <1>       ; jump from pix_op_not_w
8500                              <1>       ;
8501                              <1>       ; INPUT:
8502                              <1>       ;   ecx = bytes per row (to be applied)
8503                              <1>       ;   edx = screen width in bytes
8504                              <1>       ;   ebx = row count
8505                              <1>       ;
8506                              <1>       ;   [maskcolor] = mask color (to be excluded)
8507                              <1>       ;
8508                              <1>       ; OUTPUT:
8509                              <1>       ;     [u.r0] will be > 0 if succesful
8510                              <1>
8511                              <1>       ; window
8512                              <1>       ;mov  edi, [v_str] ; LFB start address
8513                              <1>       ;mov  esi, edi
8514                              <1>
8515 0000FEC9 803D[89120300]08    <1>       cmp   byte [v_bpp], 8 ; 8bpp
8516 0000FED0 7707                <1>       ja    short m_pix_op_not_w_1
8517                              <1>
8518                              <1>       ; 256 colors (8bpp)
8519 0000FED2 BD[4DFE0000]        <1>       mov   ebp, m_pix_op_not_8
8520 0000FED7 EB1E                <1>       jmp   short m_pix_op_not_w_4
8521                              <1>
8522                              <1> m_pix_op_not_w_1:
8523 0000FED9 803D[89120300]18    <1>       cmp   byte [v_bpp], 24 ; 24bpp
8524 0000FEE0 7710                <1>       ja    short m_pix_op_not_w_3 ; 32bpp
8525 0000FEE2 7207                <1>       jb    short m_pix_op_not_w_2 ; 16bpp
8526                              <1>
8527                              <1>       ; 24 bit true colors
8528 0000FEE4 BD[6DFE0000]        <1>       mov   ebp, m_pix_op_not_24
8529 0000FEE9 EB0C                <1>       jmp   short m_pix_op_not_w_4
8530                              <1>
8531                              <1>       ; 65536 colors (16bpp)
8532                              <1> m_pix_op_not_w_2:
8533 0000FEEB BD[96FE0000]        <1>       mov   ebp, m_pix_op_not_16
8534 0000FEF0 EB05                <1>       jmp   short m_pix_op_not_w_4
8535                              <1>
8536                              <1>       ; 32 bit true colors
8537                              <1> m_pix_op_not_w_3:
8538 0000FEF2 BD[B1FE0000]        <1>       mov   ebp, m_pix_op_not_32
8539                              <1> m_pix_op_not_w_4:
8540 0000FEF7 E95FFAFFFF          <1>       jmp   m_pix_op_not_w_x
```

```
                                <1>
                                <1> m_pix_op_neg:
                                <1>       ; 06/02/2021
                                <1>       ; NEGATIVE COLOR (MASKED, full screen)
                                <1>       ;
                                <1>       ; jump from pix_op_neg
                                <1>       ;
                                <1>       ; INPUT:
                                <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_neg_0:
0000FEFC 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0000FF03 7715                   <1>       ja     short m_pix_op_neg_1
                                <1>       ; 256 colors (8bpp)
                                <1>       ;jmp   short m_pix_op_neg_8
                                <1> m_pix_op_neg_8:
                                <1>       ; 8 bit colors (256 colors)
0000FF05 AC                     <1>       lodsb
0000FF06 3A05[9A120300]         <1>       cmp    al, [maskcolor]
0000FF0C 7408                   <1>       je     short m_pix_op_neg_8_1 ; exclude
0000FF0E F61F                   <1>       neg    byte [edi]
0000FF10 FF05[64030300]         <1>       inc    dword [u.r0] ; +1
                                <1> m_pix_op_neg_8_1:
0000FF16 47                     <1>       inc    edi
0000FF17 E2EC                   <1>       loop   m_pix_op_neg_8
0000FF19 C3                     <1>       retn
                                <1> m_pix_op_neg_1:
0000FF1A 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0000FF21 7746                   <1>       ja     short m_pix_op_neg_3 ; 32bpp
0000FF23 7229                   <1>       jb     short m_pix_op_neg_2 ; 16bpp
                                <1>       ; 24 bit true colors
                                <1>       ;jmp   short m_pix_op_neg_24
                                <1> m_pix_op_neg_24:
                                <1>       ; 24 bit true colors
0000FF25 66AD                   <1>       lodsw
0000FF27 C1E010                 <1>       shl    eax, 16
0000FF2A AC                     <1>       lodsb
0000FF2B C1C010                 <1>       rol    eax, 16
0000FF2E 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FF34 7412                   <1>       je     short m_pix_op_neg_24_1 ; exclude
0000FF36 F7D8                   <1>       neg    eax
0000FF38 668907                 <1>       mov    [edi], ax
0000FF3B C1E810                 <1>       shr    eax, 16
0000FF3E 884702                 <1>       mov    [edi+2], al
0000FF41 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_neg_24_1:
0000FF48 83C703                 <1>       add    edi, 3 ; +3
0000FF4B E2D8                   <1>       loop   m_pix_op_neg_24
0000FF4D C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_neg_2:
                                <1>       ;jmp   short m_pix_op_neg_16
                                <1> m_pix_op_neg_16:
                                <1>       ; 16 bit colors (65536 colors)
0000FF4E 66AD                   <1>       lodsw
0000FF50 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
0000FF57 740A                   <1>       je     short m_pix_op_neg_16_1 ; exclude
0000FF59 66F71F                 <1>       neg    word [edi]
0000FF5C 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
                                <1> m_pix_op_neg_16_1:
0000FF63 83C702                 <1>       add    edi, 2 ; +2
0000FF66 E2E6                   <1>       loop   m_pix_op_neg_16
0000FF68 C3                     <1>       retn
                                <1> m_pix_op_neg_3:
                                <1>       ; 32 bit true colors
                                <1>       ;jmp   short m_pix_op_neg_32
                                <1> m_pix_op_neg_32:
                                <1>       ; 32 bit true colors
0000FF69 AD                     <1>       lodsd
0000FF6A 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
0000FF70 7409                   <1>       je     short m_pix_op_neg_32_1 ; exclude
0000FF72 F71F                   <1>       neg    dword [edi]
0000FF74 8305[64030300]04       <1>       add    dword [u.r0], 4 ; +4
                                <1> m_pix_op_neg_32_1:
0000FF7B 83C704                 <1>       add    edi, 4 ; +4
0000FF7E E2E9                   <1>       loop   m_pix_op_neg_32
0000FF80 C3                     <1>       retn
                                <1>
                                <1> m_pix_op_neg_w:
                                <1>       ; 06/02/2021
                                <1>       ; NEGATIVE COLOR (MASKED, window)
                                <1>       ;
                                <1>       ; jump from pix_op_neg_w
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;   ecx = bytes per row (to be applied)
                                <1>       ;   edx = screen width in bytes
                                <1>       ;   ebx = row count
                                <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>
```

```
8646 0000FF81 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8647 0000FF88 7707                <1>       ja     short m_pix_op_neg_w_1
8648                              <1>
8649                              <1>       ; 256 colors (8bpp)
8650 0000FF8A BD[05FF0000]        <1>       mov    ebp, m_pix_op_neg_8
8651 0000FF8F EB1E                <1>       jmp    short m_pix_op_neg_w_4
8652                              <1>
8653                              <1> m_pix_op_neg_w_1:
8654 0000FF91 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8655 0000FF98 7710                <1>       ja     short m_pix_op_neg_w_3 ; 32bpp
8656 0000FF9A 7207                <1>       jb     short m_pix_op_neg_w_2 ; 16bpp
8657                              <1>
8658                              <1>       ; 24 bit true colors
8659 0000FF9C BD[25FF0000]        <1>       mov    ebp, m_pix_op_neg_24
8660 0000FFA1 EB0C                <1>       jmp    short m_pix_op_neg_w_4
8661                              <1>
8662                              <1>       ; 65536 colors (16bpp)
8663                              <1> m_pix_op_neg_w_2:
8664 0000FFA3 BD[4EFF0000]        <1>       mov    ebp, m_pix_op_neg_16
8665 0000FFA8 EB05                <1>       jmp    short m_pix_op_neg_w_4
8666                              <1>
8667                              <1>       ; 32 bit true colors
8668                              <1> m_pix_op_neg_w_3:
8669 0000FFAA BD[69FF0000]        <1>       mov    ebp, m_pix_op_neg_32
8670                              <1> m_pix_op_neg_w_4:
8671 0000FFAF E9A7F9FFFF          <1>       jmp    m_pix_op_neg_w_x
8672                              <1>
8673                              <1> m_pix_op_inc:
8674                              <1>       ; 06/02/2021
8675                              <1>       ; INCREASE COLOR (MASKED, full screen)
8676                              <1>       ;
8677                              <1>       ; jump from pix_op_inc
8678                              <1>       ;
8679                              <1>       ; INPUT:
8680                              <1>       ;   ecx = [v_siz] ; display page pixel count
8681                              <1>       ;   esi = edi = [v_mem] ; LFB start address
8682                              <1>       ;
8683                              <1>       ;   [maskcolor] = mask color (to be excluded)
8684                              <1>       ;
8685                              <1>       ; OUTPUT:
8686                              <1>       ;      [u.r0] will be > 0 if succesful
8687                              <1>
8688                              <1>       ; Full screen
8689                              <1> m_pix_op_inc_0:
8690 0000FFB4 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8691 0000FFBB 7719                <1>       ja     short m_pix_op_inc_1
8692                              <1>       ; 256 colors (8bpp)
8693                              <1>       ;jmp   short m_pix_op_inc_8
8694                              <1> m_pix_op_inc_8:
8695                              <1>       ; 8 bit colors (256 colors)
8696 0000FFBD AC                  <1>       lodsb
8697 0000FFBE 3A05[9A120300]      <1>       cmp    al, [maskcolor]
8698 0000FFC4 740C                <1>       je     short m_pix_op_inc_8_1 ; exclude
8699 0000FFC6 FE07                <1>       inc    byte [edi]
8700 0000FFC8 7502                <1>       jnz    short m_pix_op_inc_8_0
8701 0000FFCA FE0F                <1>       dec    byte [edi]
8702                              <1> m_pix_op_inc_8_0:
8703 0000FFCC FF05[64030300]      <1>       inc    dword [u.r0] ; +1
8704                              <1> m_pix_op_inc_8_1:
8705 0000FFD2 47                  <1>       inc    edi
8706 0000FFD3 E2E8                <1>       loop   m_pix_op_inc_8
8707 0000FFD5 C3                  <1>       retn
8708                              <1> m_pix_op_inc_1:
8709 0000FFD6 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8710 0000FFDD 7752                <1>       ja     short m_pix_op_inc_3 ; 32bpp
8711 0000FFDF 7230                <1>       jb     short m_pix_op_inc_2 ; 16bpp
8712                              <1>       ; 24 bit true colors
8713                              <1>       ;jmp   short m_pix_op_inc_24
8714                              <1> m_pix_op_inc_24:
8715                              <1>       ; 24 bit true colors
8716 0000FFE1 66AD                <1>       lodsw
8717 0000FFE3 C1E010              <1>       shl    eax, 16
8718 0000FFE6 AC                  <1>       lodsb
8719 0000FFE7 C1C010              <1>       rol    eax, 16
8720 0000FFEA 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
8721 0000FFF0 7419                <1>       je     short m_pix_op_inc_24_1 ; exclude
8722 0000FFF2 40                  <1>       inc    eax
8723 0000FFF3 3DFFFFFF00          <1>       cmp    eax, 0FFFFFFh
8724 0000FFF8 7601                <1>       jna    short m_pix_op_inc_24_0
8725 0000FFFA 48                  <1>       dec    eax
8726                              <1> m_pix_op_inc_24_0:
8727 0000FFFB 668907              <1>       mov    [edi], ax
8728 0000FFFE C1E810              <1>       shr    eax, 16
8729 00010001 884702              <1>       mov    [edi+2], al
8730 00010004 8305[64030300]03    <1>       add    dword [u.r0], 3 ; +3
8731                              <1> m_pix_op_inc_24_1:
8732 0001000B 83C703              <1>       add    edi, 3 ; +3
8733 0001000E E2D1                <1>       loop   m_pix_op_inc_24
8734 00010010 C3                  <1>       retn
8735                              <1>       ; 65536 colors (16bpp)
8736                              <1> m_pix_op_inc_2:
8737                              <1>       ;jmp   short m_pix_op_inc_16
8738                              <1> m_pix_op_inc_16:
8739                              <1>       ; 16 bit colors (65536 colors)
8740 00010011 66AD                <1>       lodsw
8741 00010013 663B05[9A120300]    <1>       cmp    ax, [maskcolor]
8742 0001001A 740F                <1>       je     short m_pix_op_inc_16_1 ; exclude
8743 0001001C 66FF07              <1>       inc    word [edi]
8744 0001001F 7503                <1>       jnz    short m_pix_op_inc_16_0
8745 00010021 66FF0F              <1>       dec    word [edi]
8746                              <1> m_pix_op_inc_16_0:
8747 00010024 8305[64030300]02    <1>       add    dword [u.r0], 2 ; +2
8748                              <1> m_pix_op_inc_16_1:
8749 0001002B 83C702              <1>       add    edi, 2 ; +2
8750 0001002E E2E1                <1>       loop   m_pix_op_inc_16
```

```
8751 00010030 C3                  <1>       retn
8752                              <1> m_pix_op_inc_3:
8753                              <1>       ; 32 bit true colors
8754                              <1>       ;jmp   short m_pix_op_inc_32
8755                              <1> m_pix_op_inc_32:
8756                              <1>       ; 32 bit true colors
8757 00010031 AD                  <1>       lodsd
8758 00010032 3B05[9A120300]      <1>       cmp    eax, [maskcolor]
8759 00010038 740D                <1>       je     short m_pix_op_inc_32_1 ; exclude
8760 0001003A FF07                <1>       inc    dword [edi]
8761 0001003C 7502                <1>       jnz    short m_pix_op_inc_32_0
8762 0001003E FF0F                <1>       dec    dword [edi]
8763                              <1> m_pix_op_inc_32_0:
8764 00010040 8305[64030300]04    <1>       add    dword [u.r0], 4 ; +4
8765                              <1> m_pix_op_inc_32_1:
8766 00010047 83C704              <1>       add    edi, 4 ; +4
8767 0001004A E2E5                <1>       loop   m_pix_op_inc_32
8768 0001004C C3                  <1>       retn
8769                              <1>
8770                              <1> m_pix_op_inc_w:
8771                              <1>       ; 06/02/2021
8772                              <1>       ; INCREASE COLOR (MASKED, window)
8773                              <1>       ;
8774                              <1>       ; jump from pix_op_inc_w
8775                              <1>       ;
8776                              <1>       ; INPUT:
8777                              <1>       ;   ecx = bytes per row (to be applied)
8778                              <1>       ;   edx = screen width in bytes
8779                              <1>       ;   ebx = row count
8780                              <1>       ;
8781                              <1>       ;   [maskcolor] = mask color (to be excluded)
8782                              <1>       ;
8783                              <1>       ; OUTPUT:
8784                              <1>       ;      [u.r0] will be > 0 if succesful
8785                              <1>
8786                              <1>       ; window
8787                              <1>       ;mov   edi, [v_str] ; LFB start address
8788                              <1>       ;mov   esi, edi
8789                              <1>
8790 0001004D 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8791 00010054 7707                <1>       ja     short m_pix_op_inc_w_1
8792                              <1>
8793                              <1>       ; 256 colors (8bpp)
8794 00010056 BD[BDFF0000]        <1>       mov    ebp, m_pix_op_inc_8
8795 0001005B EB1E                <1>       jmp    short m_pix_op_inc_w_4
8796                              <1>
8797                              <1> m_pix_op_inc_w_1:
8798 0001005D 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8799 00010064 7710                <1>       ja     short m_pix_op_inc_w_3 ; 32bpp
8800 00010066 7207                <1>       jb     short m_pix_op_inc_w_2 ; 16bpp
8801                              <1>
8802                              <1>       ; 24 bit true colors
8803 00010068 BD[E1FF0000]        <1>       mov    ebp, m_pix_op_inc_24
8804 0001006D EB0C                <1>       jmp    short m_pix_op_inc_w_4
8805                              <1>
8806                              <1>       ; 65536 colors (16bpp)
8807                              <1> m_pix_op_inc_w_2:
8808 0001006F BD[11000100]        <1>       mov    ebp, m_pix_op_inc_16
8809 00010074 EB05                <1>       jmp    short m_pix_op_inc_w_4
8810                              <1>
8811                              <1>       ; 32 bit true colors
8812                              <1> m_pix_op_inc_w_3:
8813 00010076 BD[31000100]        <1>       mov    ebp, m_pix_op_inc_32
8814                              <1> m_pix_op_inc_w_4:
8815 0001007B E9DBF8FFFF          <1>       jmp    m_pix_op_inc_w_x
8816                              <1>
8817                              <1> m_pix_op_dec:
8818                              <1>       ; 06/02/2021
8819                              <1>       ; DECREASE COLOR (MASKED, full screen)
8820                              <1>       ;
8821                              <1>       ; jump from pix_op_dec
8822                              <1>       ;
8823                              <1>       ; INPUT:
8824                              <1>       ;   ecx = [v_siz] ; display page pixel count
8825                              <1>       ;   esi = edi = [v_mem] ; LFB start address
8826                              <1>       ;
8827                              <1>       ;   [maskcolor] = mask color (to be excluded)
8828                              <1>       ;
8829                              <1>       ; OUTPUT:
8830                              <1>       ;      [u.r0] will be > 0 if succesful
8831                              <1>
8832                              <1>       ; Full screen
8833                              <1> m_pix_op_dec_0:
8834 00010080 803D[89120300]08    <1>       cmp    byte [v_bpp], 8 ; 8bpp
8835 00010087 7719                <1>       ja     short m_pix_op_dec_1
8836                              <1>       ; 256 colors (8bpp)
8837                              <1>       ;jmp   short m_pix_op_dec_8
8838                              <1> m_pix_op_dec_8:
8839                              <1>       ; 8 bit colors (256 colors)
8840 00010089 AC                  <1>       lodsb
8841 0001008A 3A05[9A120300]      <1>       cmp    al, [maskcolor]
8842 00010090 740C                <1>       je     short m_pix_op_dec_8_1 ; exclude
8843 00010092 FE0F                <1>       dec    byte [edi]
8844 00010094 7902                <1>       jns    short m_pix_op_dec_8_0
8845 00010096 FE07                <1>       inc    byte [edi]
8846                              <1> m_pix_op_dec_8_0:
8847 00010098 FF05[64030300]      <1>       inc    dword [u.r0] ; +1
8848                              <1> m_pix_op_dec_8_1:
8849 0001009E 47                  <1>       inc    edi
8850 0001009F E2E8                <1>       loop   m_pix_op_dec_8
8851 000100A1 C3                  <1>       retn
8852                              <1> m_pix_op_dec_1:
8853 000100A2 803D[89120300]18    <1>       cmp    byte [v_bpp], 24 ; 24bpp
8854 000100A9 774D                <1>       ja     short m_pix_op_dec_3 ; 32bpp
8855 000100AB 722B                <1>       jb     short m_pix_op_dec_2 ; 16bpp
```

```
                                <1>       ; 24 bit true colors
                                <1>       ;jmp   short m_pix_op_dec_24
                                <1> m_pix_op_dec_24:
                                <1>       ; 24 bit true colors
000100AD 66AD                   <1>       lodsw
000100AF C1E010                 <1>       shl    eax, 16
000100B2 AC                     <1>       lodsb
000100B3 C1C010                 <1>       rol    eax, 16
000100B6 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
000100BC 7414                   <1>       je     short m_pix_op_dec_24_1 ; exclude
000100BE 48                     <1>       dec    eax
000100BF 7901                   <1>       jns    short m_pix_op_dec_24_0
000100C1 40                     <1>       inc    eax
                                <1> m_pix_op_dec_24_0:
000100C2 668907                 <1>       mov    [edi], ax
000100C5 C1E810                 <1>       shr    eax, 16
000100C8 884702                 <1>       mov    [edi+2], al
000100CB 8305[64030300]03       <1>       add    dword [u.r0], 3 ; +3
                                <1> m_pix_op_dec_24_1:
000100D2 83C703                 <1>       add    edi, 3 ; +3
000100D5 E2D6                   <1>       loop   m_pix_op_dec_24
000100D7 C3                     <1>       retn
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_dec_2:
                                <1>       ;jmp   short m_pix_op_dec_16
                                <1> m_pix_op_dec_16:
                                <1>       ; 16 bit colors (65536 colors)
000100D8 66AD                   <1>       lodsw
000100DA 663B05[9A120300]       <1>       cmp    ax, [maskcolor]
000100E1 740F                   <1>       je     short m_pix_op_dec_16_1 ; exclude
000100E3 66FF0F                 <1>       dec    word [edi]
000100E6 7903                   <1>       jns    short m_pix_op_dec_16_0
000100E8 66FF07                 <1>       inc    word [edi]
                                <1> m_pix_op_dec_16_0:
000100EB 8305[64030300]02       <1>       add    dword [u.r0], 2 ; +2
                                <1> m_pix_op_dec_16_1:
000100F2 83C702                 <1>       add    edi, 2 ; +2
000100F5 E2E1                   <1>       loop   m_pix_op_dec_16
000100F7 C3                     <1>       retn
                                <1> m_pix_op_dec_3:
                                <1>       ; 32 bit true colors
                                <1>       ;jmp   short m_pix_op_dec_32
                                <1> m_pix_op_dec_32:
                                <1>       ; 32 bit true colors
000100F8 AD                     <1>       lodsd
000100F9 3B05[9A120300]         <1>       cmp    eax, [maskcolor]
000100FF 740D                   <1>       je     short m_pix_op_dec_32_1 ; exclude
00010101 FF0F                   <1>       dec    dword [edi]
00010103 7902                   <1>       jns    short m_pix_op_dec_32_0
00010105 FF07                   <1>       inc    dword [edi]
                                <1> m_pix_op_dec_32_0:
00010107 8305[64030300]04       <1>       add    dword [u.r0], 4 ; +4
                                <1> m_pix_op_dec_32_1:
0001010E 83C704                 <1>       add    edi, 4 ; +4
00010111 E2E5                   <1>       loop   m_pix_op_dec_32
00010113 C3                     <1>       retn
                                <1>
                                <1> m_pix_op_dec_w:
                                <1>       ; 06/02/2021
                                <1>       ; DECREASE COLOR (MASKED, window)
                                <1>       ;
                                <1>       ; jump from pix_op_dec_w
                                <1>       ;
                                <1>       ; INPUT:
                                <1>       ;   ecx = bytes per row (to be applied)
                                <1>       ;   edx = screen width in bytes
                                <1>       ;   ebx = row count
                                <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>
00010114 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
0001011B 7707                   <1>       ja     short m_pix_op_dec_w_1
                                <1>
                                <1>       ; 256 colors (8bpp)
0001011D BD[89000100]           <1>       mov    ebp, m_pix_op_dec_8
00010122 EB1E                   <1>       jmp    short m_pix_op_dec_w_4
                                <1>
                                <1> m_pix_op_dec_w_1:
00010124 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
0001012B 7710                   <1>       ja     short m_pix_op_dec_w_3 ; 32bpp
0001012D 7207                   <1>       jb     short m_pix_op_dec_w_2 ; 16bpp
                                <1>
                                <1>       ; 24 bit true colors
0001012F BD[AD000100]           <1>       mov    ebp, m_pix_op_dec_24
00010134 EB0C                   <1>       jmp    short m_pix_op_dec_w_4
                                <1>
                                <1>       ; 65536 colors (16bpp)
                                <1> m_pix_op_dec_w_2:
00010136 BD[D8000100]           <1>       mov    ebp, m_pix_op_dec_16
0001013B EB05                   <1>       jmp    short m_pix_op_dec_w_4
                                <1>
                                <1>       ; 32 bit true colors
                                <1> m_pix_op_dec_w_3:
0001013D BD[F8000100]           <1>       mov    ebp, m_pix_op_dec_32
                                <1> m_pix_op_dec_w_4:
00010142 E914F8FFFF             <1>       jmp    m_pix_op_dec_w_x
                                <1>
                                <1> sysvideo_39:
```

```
                                <1>       ; 15/02/2021
                                <1>       ; 07/02/2021, 08/02/2021
                                <1>       ; 03/01/2021, 04/01/2021
                                <1>       ; 23/11/2020
                                <1>       ; BH = 3
                                <1>       ; PIXEL READ/WRITE
                                <1>
                                <1>       ; 07/02/2021
                                <1>       ; 04/01/2021 (TRDOS 386 v2.0.3)
00010147 80FB03                 <1>       cmp    bl, 3
0001014A 761A                   <1>       jna    short sysvideo_39_1
                                <1>       ; 07/02/2021
0001014C 80FB06                 <1>       cmp    bl, 6
0001014F 7705                   <1>       ja     short sysvideo_39_0
00010151 E91A010000             <1>       jmp    sysvideo_39_31
                                <1> sysvideo_39_0:
                                <1>       ; error
00010156 B3FF                   <1>       mov    bl, 0FFh
00010158 8B2D[60030300]         <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
0001015E 895D10                 <1>       mov    [ebp+16], ebx ; EBX
00010161 E9A6D7FFFF             <1>       jmp    sysret
                                <1> sysvideo_39_1:
00010166 803D[DA6F0000]FF       <1>       cmp    byte [CRT_MODE], 0FFh
0001016D 7312                   <1>       jnb    short sysvideo_39_2 ; SVGA (VESA VBE) video mode
                                <1>
                                <1>       ; Std VGA or CGA mode
0001016F 81C200000A00           <1>       add    edx, 0A0000h
00010175 72DF                   <1>       jc     short sysvideo_39_0
00010177 81FAFFFF0A00           <1>       cmp    edx, 0AFFFFh
0001017D 77D7                   <1>       ja     short sysvideo_39_0
0001017F EB1E                   <1>       jmp    short sysvideo_39_3 ; 8bpp
                                <1>
                                <1> sysvideo_39_2:
                                <1>       ; use current vbe (svga) video mode
                                <1>
                                <1>       ; get LFB address
00010181 A1[2C120300]           <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
00010186 09C0                   <1>       or     eax, eax
00010188 74CC                   <1>       jz     short sysvideo_39_0
0001018A 3B15[30120300]         <1>       cmp    edx, [LFB_SIZE]
00010190 73C4                   <1>       jnb    short sysvideo_39_0
                                <1>
00010192 01C2                   <1>       add    edx, eax
                                <1>       ;jc    short sysvideo_39_0
                                <1>
                                <1>       ; Pixel read/write in VESA VBE (2/3) video mode
                                <1>       ; Video memory at Linear Frame Buffer base address
                                <1>
00010194 8A3D[38120300]         <1>       mov    bh, [LFB_Info+LFBINFO.bpp]
                                <1>
0001019A 80FF08                 <1>       cmp    bh, 8 ; 8bpp
0001019D 775D                   <1>       ja     short sysvideo_39_17
                                <1>
                                <1>       ; 8 bits per pixel
                                <1> sysvideo_39_3:
0001019F 80FB01                 <1>       cmp    bl, 1 ; 1 = write pixel
000101A2 7406                   <1>       je     short sysvideo_39_5
000101A4 7712                   <1>       ja     short sysvideo_39_8
                                <1> sysvideo_39_4:
                                <1>       ; read pixel (8bpp)
000101A6 8A02                   <1>       mov    al, [edx]
                                <1>       ;mov   [u.r0], al
                                <1>       ;jmp   sysret
000101A8 EB04                   <1>       jmp    short sysvideo_39_7
                                <1> sysvideo_39_5:
                                <1>       ; write pixel (8bpp)
000101AA 88C8                   <1>       mov    al, cl
                                <1> sysvideo_39_6:
000101AC 8802                   <1>       mov    [edx], al
                                <1> sysvideo_39_7:
000101AE A2[64030300]           <1>       mov    [u.r0], al
000101B3 E954D7FFFF             <1>       jmp    sysret
                                <1> sysvideo_39_8:
000101B8 80FB03                 <1>       cmp    bl, 3 ; mix
000101BB 7208                   <1>       jb     short sysvideo_39_9
                                <1>       ; mix  pixel colors (8bpp)
000101BD 8A02                   <1>       mov    al, [edx]
000101BF 00C8                   <1>       add    al, cl
000101C1 D0D8                   <1>       rcr    al, 1
000101C3 EBE7                   <1>       jmp    short sysvideo_39_6
                                <1> sysvideo_39_9:
                                <1>        ; swap pixel colors (8bpp)
000101C5 88C8                   <1>       mov    al, cl
000101C7 8602                   <1>       xchg   [edx], al
000101C9 EBE3                   <1>       jmp    short sysvideo_39_7
                                <1>
                                <1>       ; 16 bits per pixel
                                <1> sysvideo_39_10:
000101CB 80FB01                 <1>       cmp    bl, 1 ; 1 = write pixel
000101CE 7406                   <1>       je     short sysvideo_39_12
000101D0 7714                   <1>       ja     short sysvideo_39_15
                                <1> sysvideo_39_11:
                                <1>       ; read pixel (16bpp)
000101D2 8B02                   <1>       mov    eax, [edx]
                                <1>       ;mov   [u.r0], ax
                                <1>       ;jmp   sysret
000101D4 EB05                   <1>       jmp    short sysvideo_39_14
                                <1> sysvideo_39_12:
                                <1>       ; write pixel (16bpp)
000101D6 89C8                   <1>       mov    eax, ecx
                                <1> sysvideo_39_13:
000101D8 668902                 <1>       mov    [edx], ax
                                <1> sysvideo_39_14:
000101DB 66A3[64030300]         <1>       mov    [u.r0], ax
000101E1 E926D7FFFF             <1>       jmp    sysret
```

```
                                <1> sysvideo_39_15:
000101E6 80FB03                 <1>         cmp    bl, 3 ; mix
000101E9 720A                   <1>         jb     short sysvideo_39_16
                                <1>         ; mix  pixel colors (16bpp)
000101EB 8B02                   <1>         mov    eax, [edx]
000101ED 6601C8                 <1>         add    ax, cx
000101F0 66D1D8                 <1>         rcr    ax, 1
000101F3 EBE3                   <1>         jmp    short sysvideo_39_13
                                <1> sysvideo_39_16:
                                <1>         ; swap pixel colors (16bpp)
000101F5 89C8                   <1>         mov    eax, ecx
000101F7 668702                 <1>         xchg   [edx], ax
000101FA EBDF                   <1>         jmp    short sysvideo_39_14
                                <1> sysvideo_39_17:
000101FC 80FF18                 <1>         cmp    bh, 24
000101FF 7743                   <1>         ja     short sysvideo_39_24
00010201 72C8                   <1>         jb     short sysvideo_39_10
                                <1>
                                <1>         ; 24 bits per pixel
00010203 81E1FFFFFF00           <1>         and    ecx, 0FFFFFFh
00010209 80FB01                 <1>         cmp    bl, 1 ; 1 = write pixel
0001020C 7406                   <1>         je     short sysvideo_39_19
0001020E 7712                   <1>         ja     short sysvideo_39_22
                                <1> sysvideo_39_18:
                                <1>         ; read pixel (24bpp)
00010210 8B02                   <1>         mov    eax, [edx]
                                <1>         ;and   eax, 0FFFFFFh
                                <1>         ;mov   [u.r0], eax
                                <1>         ;jmp   sysret
00010212 EB04                   <1>         jmp    short sysvideo_39_21
                                <1> sysvideo_39_19:
                                <1>         ; write pixel (24bpp)
00010214 89C8                   <1>         mov    eax, ecx
                                <1> sysvideo_39_20:
                                <1>         ;and   eax, 0FFFFFFh
00010216 8902                   <1>         mov    [edx], eax
                                <1> sysvideo_39_21:
00010218 A3[64030300]           <1>         mov    [u.r0], eax
0001021D E9EAD6FFFF             <1>         jmp    sysret
                                <1> sysvideo_39_22:
00010222 80FB03                 <1>         cmp    bl, 3 ; mix
00010225 720D                   <1>         jb     short sysvideo_39_23
                                <1>         ; mix  pixel colors (24bpp)
00010227 8B02                   <1>         mov    eax, [edx]
00010229 25FFFFFF00             <1>         and    eax, 0FFFFFFh
                                <1>         ;and   ecx, 0FFFFFFh
0001022E 01C8                   <1>         add    eax, ecx
00010230 D1D8                   <1>         rcr    eax, 1
00010232 EBE2                   <1>         jmp    short sysvideo_39_20
                                <1> sysvideo_39_23:
                                <1>         ; swap pixel colors (24bpp)
00010234 89C8                   <1>         mov    eax, ecx
                                <1>         ;and   eax, 0FFFFFFh
00010236 668702                 <1>         xchg   [edx], ax
00010239 C1C810                 <1>         ror    eax, 16
0001023C 884202                 <1>         mov    [edx+2], al
0001023F C1C010                 <1>         rol    eax, 16
00010242 EBD4                   <1>         jmp    short sysvideo_39_21
                                <1>
                                <1>         ; 32 bits per pixel
                                <1> sysvideo_39_24:
00010244 80FB01                 <1>         cmp    bl, 1 ; 1 = write pixel
00010247 7406                   <1>         je     short sysvideo_39_26
00010249 7712                   <1>         ja     short sysvideo_39_29
                                <1> sysvideo_39_25:
                                <1>         ; read pixel (32bpp)
0001024B 8B02                   <1>         mov    eax, [edx]
                                <1>         ;mov   [u.r0], eax
                                <1>         ;jmp   sysret
0001024D EB04                   <1>         jmp    short sysvideo_39_28
                                <1> sysvideo_39_26:
                                <1>         ; write pixel (32bpp)
0001024F 89C8                   <1>         mov    eax, ecx
                                <1> sysvideo_39_27:
00010251 8902                   <1>         mov    [edx], eax
                                <1> sysvideo_39_28:
00010253 A3[64030300]           <1>         mov    [u.r0], eax
00010258 E9AFD6FFFF             <1>         jmp    sysret
                                <1> sysvideo_39_29:
0001025D 80FB03                 <1>         cmp    bl, 3 ; mix
00010260 7208                   <1>         jb     short sysvideo_39_30
                                <1>         ; mix  pixel colors (32bpp)
00010262 8B02                   <1>         mov    eax, [edx]
00010264 01C8                   <1>         add    eax, ecx
00010266 D1D8                   <1>         rcr    eax, 1
00010268 EBE7                   <1>         jmp    short sysvideo_39_27
                                <1> sysvideo_39_30:
                                <1>         ; swap pixel colors (32bpp)
0001026A 89C8                   <1>         mov    eax, ecx
0001026C 8702                   <1>         xchg   [edx], eax
0001026E EBE3                   <1>         jmp    short sysvideo_39_28
                                <1>
                                <1> sysvideo_39_31:
                                <1>         ; 08/02/2021
                                <1>         ; 07/02/2021
                                <1>         ; BL = 4 -> read pixels from user defined positions
                                <1>         ; BL = 5 -> write single color pixels to user defined pos.
                                <1>         ; BL = 6 -> write multi color pixels to user defined pos.
                                <1>         ; ECX = color (CL, CX, ECX)
                                <1>         ; EDX = number of pixels
                                <1>         ; ESI = user buffer contains dword pixel positions
                                <1>         ;       (and dword colors for BL input = 6)
                                <1>         ; EDI = user's pixel color buff (destination) for BL = 4
                                <1>
00010270 890D[9A120300]         <1>         mov    [maskcolor], ecx
```

```
9171 00010276 89D5               <1>       mov    ebp, edx ; number of pixels
9172 00010278 803D[DA6F0000]FF   <1>       cmp    byte [CRT_MODE], 0FFh ; SVGA flag
9173 0001027F 7317               <1>       jnb    short sysvideo_39_33 ; SVGA (VESA VBE mode)
9174                             <1>       ; Standard VGA mode
9175 00010281 B900000100         <1>       mov    ecx, 65536 ; Video page size (maximum)
9176 00010286 39CA               <1>       cmp    edx, ecx
9177 00010288 7709               <1>       ja     short sysvideo_39_32 ; abnormal value !
9178 0001028A B800000A00         <1>       mov    eax, 0A0000h ; Video page start address
9179 0001028F B208               <1>       mov    dl, 8  ; 8 bits per pixel (256 colors)
9180 00010291 EB20               <1>       jmp    short sysvideo_39_34
9181                             <1> sysvideo_39_32:
9182                             <1>       ; nonsense! (edx has abnormal value)
9183 00010293 E974D6FFFF         <1>       jmp    sysret
9184                             <1> sysvideo_39_33:
9185                             <1>       ; get LFB address
9186 00010298 A1[2C120300]       <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
9187 0001029D 09C0               <1>       or     eax, eax
9188 0001029F 74F2               <1>       jz     short sysvideo_39_32 ; LFB is not ready !
9189 000102A1 8B0D[30120300]     <1>       mov    ecx, [LFB_SIZE]
9190 000102A7 39CA               <1>       cmp    edx, ecx
9191 000102A9 77E8               <1>       ja     short sysvideo_39_32 ; abnormal value !
9192                             <1>
9193                             <1>       ; 08/02/2021
9194 000102AB 89D5               <1>       mov    ebp, edx ; pixel count
9195                             <1>       ;shl   ebp, 2 ; byte count (pixel pos: 4 bytes)
9196                             <1>
9197                             <1>       ; bits per pixel (pixel color size)
9198 000102AD 8A15[38120300]     <1>       mov    dl, [LFB_Info+LFBINFO.bpp]
9199                             <1> sysvideo_39_34:
9200 000102B3 C1E502             <1>       shl    ebp, 2 ; 15/02/2021 (byte count)
9201 000102B6 A3[8A120300]       <1>       mov    [v_mem], eax ; Save video page start address
9202                             <1>       ;mov   [v_siz], ecx ; Video page size (limit)
9203                             <1>       ;mov   ebx, [v_siz]
9204 000102BB 88DE               <1>       mov    dh, bl ; sub function
9205 000102BD 89CB               <1>       mov    ebx, ecx ; [v_siz]
9206 000102BF 8815[89120300]     <1>       mov    [v_bpp], dl ; bits per pixel (color size)
9207                             <1>
9208 000102C5 B900080000         <1>       mov    ecx, 2048
9209 000102CA 39CD               <1>       cmp    ebp, ecx
9210 000102CC 7302               <1>       jnb    short sysvideo_39_35
9211 000102CE 89E9               <1>       mov    ecx, ebp ; fix to requested byte count
9212                             <1> sysvideo_39_35:
9213 000102D0 80FE04             <1>       cmp    dh, 4 ; 08/02/2021
9214                             <1>       ;cmp   bl, 4 ; read pixels from user defined positions
9215 000102D3 7605               <1>       jna    short sysvideo_39_36
9216 000102D5 E9B2000000         <1>       jmp    sysvideo_39_52
9217                             <1>       ; 08/02/2021
9218                             <1>       ;mov   [buffer8], edi  ; user's destination buff addr
9219                             <1> sysvideo_39_36:
9220                             <1>       ; 08/02/2021
9221                             <1>       ; read pixel positions
9222                             <1>       ; as defined in user's source buffer
9223 000102DA 893D[A2120300]     <1>       mov    [buffer8], edi  ; user's destination buff addr
9224 000102E0 BF00760900         <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
9225                             <1>                                     ; 2028 byte data
9226                             <1>       ; esi = user's source buffer for pixel positions
9227                             <1>       ; ecx = byte count
9228 000102E5 E8A7180000         <1>       call   transfer_from_user_buffer
9229 000102EA 72A7               <1>       jc     short sysvideo_39_32 ; error
9230                             <1>       ; ecx  = transfer count (bytes)
9231                             <1>
9232 000102EC 57                 <1>       push   edi ; *
9233 000102ED 56                 <1>       push   esi ; **
9234 000102EE 51                 <1>       push   ecx ; ***
9235                             <1>
9236 000102EF 89FE               <1>       mov    esi, edi ; kernel buffer
9237 000102F1 8B15[8A120300]     <1>       mov    edx, [v_mem] ; video memory
9238 000102F7 C1E902             <1>       shr    ecx, 2 ; pixel count (within buffer capacity)
9239                             <1>
9240 000102FA 803D[89120300]08   <1>       cmp    byte [v_bpp], 8 ; 8bpp
9241 00010301 7753               <1>       ja     short sysvideo_39_49
9242                             <1> sysvideo_39_37:
9243                             <1>       ; 8bpp
9244 00010303 AD                 <1>       lodsd
9245 00010304 39D8               <1>       cmp    eax, ebx ; < [v_siz]
9246 00010306 7309               <1>       jnb    short sysvideo_39_39
9247 00010308 0FB60402           <1>       movzx  eax, byte [edx+eax]
9248                             <1> sysvideo_39_38:
9249 0001030C AB                 <1>       stosd
9250 0001030D E2F4               <1>       loop   sysvideo_39_37
9251 0001030F EB49               <1>       jmp    short sysvideo_39_50
9252                             <1> sysvideo_39_39:
9253                             <1>       ; write black color for improper positions
9254 00010311 31C0               <1>       xor    eax, eax
9255 00010313 EBF7               <1>       jmp    short sysvideo_39_38
9256                             <1> sysvideo_39_40:
9257 00010315 803D[89120300]18   <1>       cmp    byte [v_bpp], 24 ; 24bpp
9258 0001031C 772A               <1>       ja     short sysvideo_39_47 ; 32bpp
9259 0001031E 7216               <1>       jb     short sysvideo_39_44 ; 16bpp
9260                             <1> sysvideo_39_41:
9261                             <1>       ; 24bpp
9262 00010320 AD                 <1>       lodsd
9263 00010321 39D8               <1>       cmp    eax, ebx ; < [v_siz]
9264 00010323 730D               <1>       jnb    short sysvideo_39_43
9265 00010325 8B0402             <1>       mov    eax, [edx+eax]
9266 00010328 25FFFFFF00         <1>       and    eax, 0FFFFFFh
9267                             <1> sysvideo_39_42:
9268 0001032D AB                 <1>       stosd
9269 0001032E E2F0               <1>       loop   sysvideo_39_41
9270 00010330 EB28               <1>       jmp    short sysvideo_39_50
9271                             <1> sysvideo_39_43:
9272                             <1>       ; write black color for improper positions
9273 00010332 31C0               <1>       xor    eax, eax
9274 00010334 EBF7               <1>       jmp    short sysvideo_39_42
9275                             <1> sysvideo_39_44:
```

```
                                <1>       ; 16bpp
00010336 AD                     <1>       lodsd
00010337 39D8                   <1>       cmp    eax, ebx ; < [v_siz]
00010339 7309                   <1>       jnb    short sysvideo_39_46
0001033B 0FB70402               <1>       movzx  eax, word [edx+eax]
                                <1> sysvideo_39_45:
0001033F AB                     <1>       stosd
00010340 E2F4                   <1>       loop   sysvideo_39_44
00010342 EB16                   <1>       jmp    short sysvideo_39_50
                                <1> sysvideo_39_46:
                                <1>       ; write black color for improper positions
00010344 31C0                   <1>       xor    eax, eax
00010346 EBF7                   <1>       jmp    short sysvideo_39_45
                                <1> sysvideo_39_47:
                                <1>       ; 32bpp
00010348 AD                     <1>       lodsd
00010349 39D8                   <1>       cmp    eax, ebx ; < [v_siz]
0001034B 7309                   <1>       jnb    short sysvideo_39_49
0001034D 0FB70402               <1>       movzx  eax, word [edx+eax]
                                <1> sysvideo_39_48:
00010351 AB                     <1>       stosd
00010352 E2F4                   <1>       loop   sysvideo_39_47
00010354 EB04                   <1>       jmp    short sysvideo_39_50
                                <1> sysvideo_39_49:
                                <1>       ; write black color for improper positions
00010356 31C0                   <1>       xor    eax, eax
00010358 EBF7                   <1>       jmp    short sysvideo_39_48
                                <1> sysvideo_39_50:
0001035A 59                     <1>       pop    ecx ; transfer count in bytes
0001035B 5E                     <1>       pop    esi ; ** ; kernel buffer
                                <1>       ;mov   esi, VBE3SAVERESTOREBLOCK ; kernel buffer for
                                <1>                                        ; 2028 byte data
0001035C 8B3D[A2120300]         <1>       mov    edi, [buffer8]
                                <1>       ; edi = user's destination buffer for pixel colors
                                <1>       ; ecx = byte count
00010362 E8E0170000             <1>       call   transfer_to_user_buffer
00010367 5E                     <1>       pop    esi ; *
00010368 7260                   <1>       jc     short sysvideo_39_56 ; error
                                <1>       ; ecx  = transfer count (bytes)
0001036A 89C8                   <1>       mov    eax, ecx
0001036C C1E802                 <1>       shr    eax, 2
0001036F 0105[64030300]         <1>       add    [u.r0], eax ; transfer count (in pixels)
                                <1>
00010375 29CD                   <1>       sub    ebp, ecx
00010377 7651                   <1>       jna    short sysvideo_39_56 ; completed/finished
00010379 01CE                   <1>       add    esi, ecx ; next position in source buffer
                                <1>       ;add   [buffer8], ecx ; next pos in destination buff
0001037B 01CF                   <1>       add    edi, ecx
0001037D 66B90008               <1>       mov    cx, 2048 ; new count, limit: kernel buff size
00010381 39CD                   <1>       cmp    ebp, ecx ; remain >= limit ?
00010383 7302                   <1>       jnb    short sysvideo_39_51 ; yes
00010385 89E9                   <1>       mov    ecx, ebp ; fix byte count to remain bytes
                                <1> sysvideo_39_51:
00010387 E94EFFFFFF             <1>       jmp    sysvideo_39_36
                                <1>
                                <1> sysvideo_39_52:
0001038C 80FE05                 <1>       cmp    dh, 5 ; 08/02/2021
                                <1>       ;cmp   bl, 5 ; write pixels to user defined positions
0001038F 7605                   <1>       jna    short sysvideo_39_53
00010391 E99B000000             <1>       jmp    sysvideo_39_66
                                <1> sysvideo_39_53:
                                <1>       ; single color pixel writing
00010396 BF00760900             <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
                                <1>                                        ; 2028 byte data
                                <1>       ; esi = user's source buffer for pixel positions
                                <1>       ; ecx = byte count
0001039B E8F1170000             <1>       call   transfer_from_user_buffer
000103A0 7228                   <1>       jc     short sysvideo_39_56 ; error
                                <1>       ; ecx = transfer count (bytes)
                                <1>
                                <1>       ; write pixels by using (user) defined positions
                                <1>       ; ecx = byte count (1,2,3,4 times pixel count)
                                <1>       ; edi = system buffer address
                                <1>
000103A2 56                     <1>       push   esi ; **
000103A3 51                     <1>       push   ecx ; *
                                <1>
000103A4 89FE                   <1>       mov    esi, edi
000103A6 8B3D[8A120300]         <1>       mov    edi, [v_mem]
                                <1>
                                <1>       ; 08/02/2021
000103AC C1E902                 <1>       shr    ecx, 2 ; pixel count
000103AF 8B15[9A120300]         <1>       mov    edx, [maskcolor]
                                <1>       ;mov   ebx, [v_siz]
                                <1>
000103B5 803D[89120300]08       <1>       cmp    byte [v_bpp], 8 ; 8bpp
000103BC 7711                   <1>       ja     short sysvideo_39_57
                                <1> sysvideo_39_54:
                                <1>       ; 8bpp
000103BE AD                     <1>       lodsd
000103BF 39D8                   <1>       cmp    eax, ebx ; < [v_siz]
000103C1 7303                   <1>       jnb    short sysvideo_39_55
000103C3 881407                 <1>       mov    [edi+eax], dl
                                <1> sysvideo_39_55:
000103C6 E2F6                   <1>       loop   sysvideo_39_54
000103C8 EB50                   <1>       jmp    short sysvideo_39_64
                                <1> sysvideo_39_56:
000103CA E93DD5FFFF             <1>       jmp    sysret
                                <1> sysvideo_39_57:
000103CF 803D[89120300]18       <1>       cmp    byte [v_bpp], 24 ; 24bpp
000103D6 7732                   <1>       ja     short sysvideo_39_62 ; 32bpp
000103D8 721D                   <1>       jb     short sysvideo_39_60 ; 16bpp
                                <1> sysvideo_39_58:
                                <1>       ; 24bpp
000103DA AD                     <1>       lodsd
```

```
9381 000103DB 39D8              <1>       cmp    eax, ebx ; < [v_siz]
9382 000103DD 7314              <1>       jnb    short sysvideo_39_59
9383 000103DF 881407            <1>       mov    [edi+eax], dl
9384 000103E2 40                <1>       inc    eax
9385 000103E3 C1CA08            <1>       ror    edx, 8
9386 000103E6 66891407          <1>       mov    [edi+eax], dx
9387 000103EA C1C208            <1>       rol    edx, 8
9388 000103ED FF05[64030300]    <1>       inc    dword [u.r0]
9389                            <1> sysvideo_39_59:
9390 000103F3 E2E5              <1>       loop   sysvideo_39_58
9391 000103F5 EB23              <1>       jmp    short sysvideo_39_64
9392                            <1> sysvideo_39_60:
9393                            <1>       ; 16bpp
9394 000103F7 AD                <1>       lodsd
9395 000103F8 39D8              <1>       cmp    eax, ebx ; < [v_siz]
9396 000103FA 730A              <1>       jnb    short sysvideo_39_61
9397 000103FC 66891407          <1>       mov    [edi+eax], dx
9398 00010400 FF05[64030300]    <1>       inc    dword [u.r0]
9399                            <1> sysvideo_39_61:
9400 00010406 E2EF              <1>       loop   sysvideo_39_60
9401 00010408 EB10              <1>       jmp    short sysvideo_39_64
9402                            <1> sysvideo_39_62:
9403                            <1>       ; 32bpp
9404 0001040A AD                <1>       lodsd
9405 0001040B 39D8              <1>       cmp    eax, ebx ; < [v_siz]
9406 0001040D 7309              <1>       jnb    short sysvideo_39_63
9407 0001040F 891407            <1>       mov    [edi+eax], edx
9408 00010412 FF05[64030300]    <1>       inc    dword [u.r0]
9409                            <1> sysvideo_39_63:
9410 00010418 E2F0              <1>       loop   sysvideo_39_62
9411                            <1> sysvideo_39_64:
9412 0001041A 59                <1>       pop    ecx ; **
9413 0001041B 5E                <1>       pop    esi ; *
9414 0001041C 29CD              <1>       sub    ebp, ecx
9415 0001041E 76AA              <1>       jna    short sysvideo_39_56
9416 00010420 01CE              <1>       add    esi, ecx
9417 00010422 66B90008          <1>       mov    cx, 2048
9418 00010426 39CD              <1>       cmp    ebp, ecx
9419 00010428 7302              <1>       jnb    short sysvideo_39_65
9420 0001042A 89E9              <1>       mov    ecx, ebp
9421                            <1> sysvideo_39_65:
9422 0001042C E965FFFFFF        <1>       jmp    sysvideo_39_53
9423                            <1>
9424                            <1> sysvideo_39_66:
9425                            <1>       ; 15/02/2021
9426 00010431 D1E5              <1>       shl    ebp, 1 ; 8 bytes per pixel (position&color)
9427                            <1> sysvideo_39_67:
9428 00010433 66B90008          <1>       mov    cx, 2048
9429 00010437 39CD              <1>       cmp    ebp, ecx
9430 00010439 7302              <1>       jnb    short sysvideo_39_68
9431 0001043B 89E9              <1>       mov    ecx, ebp
9432                            <1> sysvideo_39_68:
9433                            <1>       ; multi colors pixel writing
9434 0001043D BF00760900        <1>       mov    edi, VBE3SAVERESTOREBLOCK ; kernel buffer for
9435                            <1>                                      ; 2048 byte data
9436                            <1>       ; esi = user's source buffer for pixel positions
9437                            <1>       ; ecx = byte count
9438 00010442 E84A170000        <1>       call   transfer_from_user_buffer
9439 00010447 7281              <1>       jc     short sysvideo_39_56 ; error
9440                            <1>       ; ecx  = transfer count
9441                            <1>
9442                            <1>       ; write pixels & colors as defined in user buffer
9443                            <1>       ; ecx = byte count (2,4,6,8 times pixel count)
9444                            <1>       ; edi = system buffer address
9445                            <1>
9446 00010449 56                <1>       push   esi ; **
9447 0001044A 51                <1>       push   ecx ; *
9448                            <1>
9449 0001044B 89FE              <1>       mov    esi, edi
9450 0001044D 8B3D[8A120300]    <1>       mov    edi, [v_mem]
9451                            <1>
9452                            <1>       ; 08/02/2021
9453 00010453 C1E903            <1>       shr    ecx, 3 ; pixel count
9454                            <1>
9455                            <1>       ;mov   ebx, [v_siz]
9456                            <1>
9457 00010456 803D[89120300]08  <1>       cmp    byte [v_bpp], 8 ; 8bpp
9458 0001045D 770F              <1>       ja     short sysvideo_39_71
9459                            <1> sysvideo_39_69:
9460                            <1>       ; 8bpp
9461 0001045F AD                <1>       lodsd  ; position
9462 00010460 89C2              <1>       mov    edx, eax
9463 00010462 AD                <1>       lodsd  ; color
9464 00010463 39DA              <1>       cmp    edx, ebx ; < [v_siz]
9465 00010465 7303              <1>       jnb    short sysvideo_39_70
9466 00010467 880417            <1>       mov    [edi+edx], al
9467                            <1> sysvideo_39_70:
9468 0001046A E2F3              <1>       loop   sysvideo_39_69
9469 0001046C EB51              <1>       jmp    short sysvideo_39_78
9470                            <1> sysvideo_39_71:
9471 0001046E 803D[89120300]18  <1>       cmp    byte [v_bpp], 24 ; 24bpp
9472 00010475 7735              <1>       ja     short sysvideo_39_76 ; 32bpp
9473 00010477 721D              <1>       jb     short sysvideo_39_74 ; 16bpp
9474                            <1> sysvideo_39_72:
9475                            <1>       ; 24bpp
9476 00010479 AD                <1>       lodsd  ; position
9477 0001047A 89C2              <1>       mov    edx, eax
9478 0001047C AD                <1>       lodsd  ; color
9479 0001047D 39DA              <1>       cmp    edx, ebx ; < [v_siz]
9480 0001047F 7311              <1>       jnb    short sysvideo_39_73
9481 00010481 880417            <1>       mov    [edi+edx], al
9482 00010484 42                <1>       inc    edx
9483 00010485 C1E808            <1>       shr    eax, 8
9484 00010488 66890417          <1>       mov    [edi+edx], ax
9485 0001048C FF05[64030300]    <1>       inc    dword [u.r0]
```

```
                                <1> sysvideo_39_73:
00010492 E2E5                   <1>         loop    sysvideo_39_72
00010494 EB29                   <1>         jmp     short sysvideo_39_78
                                <1> sysvideo_39_74:
                                <1>         ; 16bpp
00010496 AD                     <1>         lodsd   ; position
00010497 89C2                   <1>         mov     edx, eax
00010499 AD                     <1>         lodsd   ; color
0001049A 39DA                   <1>         cmp     edx, ebx ; < [v_siz]
0001049C 730A                   <1>         jnb     short sysvideo_39_75
0001049E 66890417               <1>         mov     [edi+edx], ax
000104A2 FF05[64030300]         <1>         inc     dword [u.r0]
                                <1> sysvideo_39_75:
000104A8 E2EC                   <1>         loop    sysvideo_39_74
000104AA EB13                   <1>         jmp     short sysvideo_39_78
                                <1> sysvideo_39_76:
                                <1>         ; 32bpp
000104AC AD                     <1>         lodsd   ; position
000104AD 89C2                   <1>         mov     edx, eax
000104AF AD                     <1>         lodsd   ; color
000104B0 39DA                   <1>         cmp     edx, ebx ; < [v_siz]
000104B2 7309                   <1>         jnb     short sysvideo_39_77
000104B4 890417                 <1>         mov     [edi+edx], eax
000104B7 FF05[64030300]         <1>         inc     dword [u.r0]
                                <1> sysvideo_39_77:
000104BD E2ED                   <1>         loop    sysvideo_39_76
                                <1> sysvideo_39_78:
000104BF 59                     <1>         pop     ecx ; *
000104C0 5E                     <1>         pop     esi ; **
                                <1>
000104C1 29CD                   <1>         sub     ebp, ecx
000104C3 762E                   <1>         jna     short sysvideo_39_79
000104C5 01CE                   <1>         add     esi, ecx
000104C7 E967FFFFFF             <1>         jmp     sysvideo_39_67
                                <1> ;sysvideo_39_79:
                                <1> ;       jmp     sysret
                                <1>
                                <1> sysvideo_16:
                                <1>         ; 23/11/2020
000104CC 80FF04                 <1>         cmp     bh, 4
000104CF 0F8272FCFFFF           <1>         jb      sysvideo_39 ; bh = 3, pixel r/w
000104D5 7721                   <1>         ja      short sysvideo_17
                                <1>
                                <1>         ; BH = 4
                                <1>         ; Direct User Access for CGA video memory.
                                <1>         ; Setup user's page tables for direct access to 0B8000h.
                                <1>         ;
                                <1>         ; Permission checks are not implemented yet !
                                <1>         ; (11/07/2016)
                                <1>
000104D7 B800800B00             <1>         mov     eax, 0B8000h
000104DC B908000000             <1>         mov     ecx, 8 ; 8 pages (8*4K=32K)
000104E1 89C3                   <1>         mov     ebx, eax ; 12/05/2017 ; virtual = physical
000104E3 E81A62FFFF             <1>         call    direct_memory_access
000104E8 0F821ED4FFFF           <1>         jc      sysret
                                <1>         ; eax = 0B8000h if there is not an error
000104EE A3[64030300]           <1>         mov     [u.r0], eax
                                <1> sysvideo_39_79: ; 08/01/2021
000104F3 E914D4FFFF             <1>         jmp     sysret
                                <1>
                                <1> sysvideo_17:
                                <1>         ; 23/12/2020
                                <1>         ; 11/12/2020
                                <1>         ; 10/12/2020
                                <1>         ; 23/11/2020
000104F8 80FF06                 <1>         cmp     bh, 6
000104FB 740B                   <1>         je      short sysvideo_17_0
000104FD 0F82B9000000           <1>         jb      sysvideo_18
00010503 E913010000             <1>         jmp     sysvideo_20 ; ja
                                <1>
                                <1> sysvideo_17_0:
                                <1>         ; BH = 6
                                <1>         ; Direct User Access to Linear Frame Buffer.
                                <1>         ; Setup user's page tables for direct access to LFB.
                                <1>         ;
                                <1>         ; Permission checks are not implemented yet !
                                <1>         ; (10/12/2020)
                                <1>
00010508 80FBFF                 <1>         cmp     bl, 0FFh ; current video mode
0001050B 722C                   <1>         jb      short sysvideo_17_2 ; for desired video mode
                                <1>
0001050D 381D[DA6F0000]         <1>         cmp     [CRT_MODE], bl ; VESA VBE video mode ?
00010513 750E                   <1>         jne     short sysvideo_17_1
00010515 668B0D[1E120300]       <1>         mov     cx, [video_mode]
0001051C 6681E1FF01             <1>         and     cx, 1FFh
00010521 EB29                   <1>         jmp     short sysvideo_17_3
                                <1> sysvideo_17_1:
                                <1>         ; 11/12/2020
00010523 88DF                   <1>         mov     bh, bl ; 0FFh
00010525 8A1D[DA6F0000]         <1>         mov     bl, [CRT_MODE] ; VGA/CGA video mode
0001052B 8B2D[60030300]         <1>         mov     ebp, [u.usp]  ; ebp points to user's registers
                                <1>         ; 23/12/2020
00010531 895D10                 <1>         mov     [ebp+16], ebx ; return to user with EBX value
00010534 E9D3D3FFFF             <1>         jmp     sysret ; return to user with EAX = 0
                                <1> sysvideo_17_2:
                                <1>         ; bl = VESA video mode - 100h
00010539 B701                   <1>         mov     bh, 1 ; bx = 1XXh
0001053B 53                     <1>         push    ebx ; requested vesa video mode
0001053C E87236FFFF             <1>         call    vbe_biosfn_return_current_mode
00010541 59                     <1>         pop     ecx ; requested vesa video mode
00010542 6681E3FF01             <1>         and     bx, 1FFh
00010547 6639D9                 <1>         cmp     cx, bx
0001054A 7564                   <1>         jne     short sysvideo_17_8
                                <1> sysvideo_17_3:
0001054C 663B0D[2A120300]       <1>         cmp     cx, [LFB_Info+LFBINFO.mode]
```

```
9591 00010553 755B              <1>       jne    short sysvideo_17_8
9592                            <1> sysvideo_17_4:
9593                            <1>       ; 11/12/2020
9594 00010555 A1[2C120300]      <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
9595                            <1>       ; 21/12/2020
9596 0001055A 09C0              <1>       or     eax, eax
9597 0001055C 744D              <1>       jz     short sysvideo_17_7
9598                            <1>       ;
9599 0001055E 8B0D[30120300]    <1>       mov    ecx, [LFB_Info+LFBINFO.LFB_size] ; buff size
9600 00010564 89C3              <1>       mov    ebx, eax ; user's address = physical address
9601                            <1>       ;push  ebx
9602 00010566 51                <1>       push   ecx
9603                            <1>       ; 21/12/2020
9604 00010567 81C1FF0F0000      <1>       add    ecx, 4095  ; PAGESIZE - 1
9605                            <1>       ; 14/12/2020
9606 0001056D C1E90C            <1>       shr    ecx, 12  ; convert bytes to pages
9607 00010570 E88D61FFFF        <1>       call   direct_memory_access
9608 00010575 5A                <1>       pop    edx  ; linear frame buffer size in bytes
9609                            <1>       ;pop   eax  ; linear frame buffer address (physical)
9610 00010576 7233              <1>       jc     short sysvideo_17_7 ; [u.r0] = eax = 0
9611                            <1> sysvideo_17_5:
9612 00010578 668B0D[36120300]  <1>       mov    cx, [LFB_Info+LFBINFO.Y_res] ; screen height
9613 0001057F C1E110            <1>       shl    ecx, 16
9614 00010582 668B0D[34120300]  <1>       mov    cx,  [LFB_Info+LFBINFO.X_res] ; screen width
9615 00010589 31DB              <1>       xor    ebx, ebx
9616 0001058B 8A1D[38120300]    <1>       mov    bl, [LFB_Info+LFBINFO.bpp] ; bits per pixel
9617 00010591 8A3D[2A120300]    <1>       mov    bh, [LFB_Info+LFBINFO.mode] ; XX part of 1XXh
9618                            <1> sysvideo_26_4: ; 23/12/2020
9619 00010597 8B2D[60030300]    <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
9620 0001059D 895514            <1>       mov    [ebp+20], edx ; return to user with EDX value
9621 000105A0 895D10            <1>       mov    [ebp+16], ebx ; EBX
9622 000105A3 894D18            <1>       mov    [ebp+24], ecx ; ECX
9623                            <1> sysvideo_17_6:
9624 000105A6 A3[64030300]      <1>       mov    [u.r0], eax ; LFB address
9625                            <1> sysvideo_17_7:
9626 000105AB E95CD3FFFF        <1>       jmp    sysret
9627                            <1> sysvideo_17_8:
9628                            <1>       ; cx = mode
9629                            <1>       ; 21/12/2020
9630 000105B0 80CD40            <1>       or     ch, 40h  ; Linear frame buffer flag
9631 000105B3 E81D34FFFF        <1>       call   _vbe_biosfn_return_mode_info
9632 000105B8 72F1              <1>       jc     short sysvideo_17_7
9633 000105BA EB99              <1>       jmp    short sysvideo_17_4
9634                            <1>
9635                            <1> sysvideo_18:
9636                            <1>       ; BH = 5
9637                            <1>       ; Direct User Access for VGA video memory.
9638                            <1>       ; Setup user's page tables for direct access to 0A0000h.
9639                            <1>       ;
9640                            <1>       ; Permission checks are not implemented yet !
9641                            <1>       ; (11/07/2016)
9642                            <1>
9643 000105BC B800000A00        <1>       mov    eax, 0A0000h
9644 000105C1 B910000000        <1>       mov    ecx, 16 ; 16 pages (16*4K=64K)
9645 000105C6 89C3              <1>       mov    ebx, eax ; 12/05/2017 ; virtual = physical
9646 000105C8 E83561FFFF        <1>       call   direct_memory_access
9647 000105CD 0F8239D3FFFF      <1>       jc     sysret
9648                            <1>       ; eax = 0A0000h if there is not an error
9649 000105D3 A3[64030300]      <1>       mov    [u.r0], eax
9650 000105D8 E92FD3FFFF        <1>       jmp    sysret
9651                            <1>
9652                            <1> sysvideo_19:
9653                            <1>       ; 22/01/2021
9654                            <1>       ; 12/12/2020
9655                            <1>       ; 11/12/2020
9656                            <1>       ; 23/11/2020
9657                            <1>       ; BH = 7
9658                            <1>       ; Get (Super/Extended VGA) mode
9659                            <1>       ; and Linear Frame Buffer info.
9660                            <1>
9661                            <1>       ; 22/01/2021
9662 000105DD B3FF              <1>       mov    bl, 0FFh
9663                            <1>       ; 11/12/2020
9664                            <1>       ;cmp   byte [CRT_MODE], 0FFh ; (extended mode?)
9665                            <1>       ; 22/01/2021
9666 000105DF 381D[DA6F0000]    <1>       cmp    [CRT_MODE], bl  ; 0FFh
9667                            <1>       ;jb    sysvideo_17_1; not a VESA VBE mode
9668                            <1>       ; 12/12/2020
9669 000105E5 7305              <1>       jnb    short sysvideo_19_0
9670 000105E7 E937FFFFFF        <1>       jmp    sysvideo_17_1
9671                            <1>
9672                            <1> sysvideo_19_0:
9673 000105EC E8C235FFFF        <1>       call   vbe_biosfn_return_current_mode
9674 000105F1 6681E3FF01        <1>       and    bx, 1FFh
9675 000105F6 663B1D[2A120300]  <1>       cmp    bx, [LFB_Info+LFBINFO.mode]
9676 000105FD 7510              <1>       jne    short sysvideo_19_2
9677                            <1> sysvideo_19_1:
9678 000105FF A1[2C120300]      <1>       mov    eax, [LFB_Info+LFBINFO.LFB_addr]
9679 00010604 8B15[30120300]    <1>       mov    edx, [LFB_Info+LFBINFO.LFB_size]
9680 0001060A E969FFFFFF        <1>       jmp    sysvideo_17_5
9681                            <1> sysvideo_19_2:
9682 0001060F E8C133FFFF        <1>       call   _vbe_biosfn_return_mode_info
9683 00010614 73E9              <1>       jnc    short sysvideo_19_1
9684 00010616 E9F1D2FFFF        <1>       jmp    sysret
9685                            <1>
9686                            <1> sysvideo_20:
9687                            <1>       ; 11/12/2020
9688                            <1>       ; 23/11/2020
9689 0001061B 80FF08            <1>       cmp    bh, 8
9690 0001061E 72BD              <1>       jb     short sysvideo_19 ; video mode & lfb info
9691 00010620 0F8780000000      <1>       ja     sysvideo_21 ; 12/12/2020
9692                            <1>
9693                            <1>       ; BH = 8
9694                            <1>       ; Set (Super/Extended VGA) mode & return LFB info
9695                            <1>       ;
```

```
                                <1>
                                <1>       ; 11/12/2020
00010626 80FBFF                 <1>       cmp   bl, 0FFh  ; CGA/VGA mode ?
00010629 7318                   <1>       jnb   short sysvideo_20_1
                                <1>
                                <1>       ;xor   ah, ah
0001062B 88D8                   <1>       mov   al, bl
                                <1> sysvideo_20_0:
0001062D E83F11FFFF             <1>       call  _int10h  ; uses vbe3 pmi32 option
00010632 83F8FF                 <1>       cmp   eax, 0FFFFFFFFh ; -1
00010635 7459                   <1>       je    short sysvideo_20_3 ; error
                                <1>
                                <1>       ; 11/12/2020
                                <1>       ; alternative (it does not use vbe3 pmi32)
                                <1>       ;push  eax
                                <1>       ;call  _set_mode
                                <1>       ;pop   eax
                                <1>       ;jc    short sysvideo_20_3
                                <1>
                                <1>       ;inc   eax
00010637 FEC0                   <1>       inc   al
                                <1>       ;mov   [u.r0], ax ; video mode + 1
00010639 A2[64030300]           <1>       mov   [u.r0], al
0001063E E9C9D2FFFF             <1>       jmp   sysret
                                <1>
                                <1> sysvideo_20_1:
                                <1>       ; cx = vesa video mode
00010643 6689C8                 <1>       mov   ax, cx
00010646 663D0001               <1>       cmp   ax, 100h
0001064A 72E1                   <1>       jb    short sysvideo_20_0 ; VGA/CGA mode
0001064C 663DFF01               <1>       cmp   ax, 1FFh
                                <1>       ;ja    short sysvideo_20_4 ; not valid
00010650 773E                   <1>       ja    short sysvideo_20_3
00010652 50                     <1>       push  eax
00010653 6689C3                 <1>       mov   bx, ax
00010656 66B8024F               <1>       mov   ax, 4F02h
                                <1>
                                <1>       ; simulate _int10h (int 31h) for func 4F02h
                                <1>       ;pushfd
                                <1>       ;push  cs
                                <1>       ;push  sysvideo_20_1_retn
                                <1>       ;push  es ; *
                                <1>       ;push  ds ; **      ; SAVE WORK AND PARAMETER REGISTERS
                                <1>       ;jmp   VBE_func
                                <1> ;sysvideo_20_1_retn:
                                <1>
0001065A E81211FFFF             <1>       call  _int10h ; simulate int 10h (int 31h)
                                <1>
0001065F 6683F84F               <1>       cmp   ax, 004Fh
00010663 58                     <1>       pop   eax
00010664 752A                   <1>       jne   short sysvideo_20_3 ; error
                                <1>       ;pop   eax
00010666 40                     <1>       inc   eax
00010667 A3[64030300]           <1>       mov   [u.r0], eax ; video mode + 1
0001066C 09D2                   <1>       or    edx, edx ; is LFBINFO requested by user ?
                                <1>       ;jz    short sysvideo_20_4
0001066E 7420                   <1>       jz    short sysvideo_20_3 ; no
                                <1>
                                <1>       ; 11/12/2020
                                <1>       ; Check LFBINFO table/structure
                                <1>       ; (it is set by vbe2 'vbe_biosfn_set_mode'
                                <1>       ; but if vbe3 vbios pmi is in use,
                                <1>       ; it will not set LFBINFO table)
                                <1>
00010670 52                     <1>       push  edx
00010671 48                     <1>       dec   eax  ; video mode
00010672 BE[2A120300]           <1>       mov   esi, LFB_Info
00010677 663B06                 <1>       cmp   ax, [esi+LFBINFO.mode]
0001067A 7407                   <1>       je    short sysvideo_20_2
                                <1>
0001067C E85433FFFF             <1>       call  _vbe_biosfn_return_mode_info
                                <1>       ;jnc   short sysvideo_20_2
00010681 7212                   <1>       jc    short sysvideo_20_4 ; edx = 0
                                <1>
                                <1>       ;; clear LFBINFO table for invalidating
                                <1>       ;mov   ecx, LFBINFO.size ; 16
                                <1>       ;mov   edi, esi ; LFB_Info table address
                                <1>       ;xor   eax, eax
                                <1>       ;rep   stosb
                                <1>
                                <1> sysvideo_20_2:
                                <1>       ;pop   ecx
                                <1>       ;mov   edi, ecx ; user buffer
00010683 5F                     <1>       pop   edi
00010684 B910000000             <1>       mov   ecx, LFBINFO.size ; 16
00010689 E8B9140000             <1>       call  transfer_to_user_buffer ; fast transfer
0001068E 7206                   <1>       jc    short sysvideo_20_5
                                <1>
                                <1>       ;jmp   sysret
                                <1> sysvideo_20_3:
                                <1>       ;pop   eax ; [u.r0] = 0
                                <1> ;sysvideo_20_4:
00010690 E977D2FFFF             <1>       jmp   sysret
                                <1>
                                <1> sysvideo_20_4:
00010695 5A                     <1>       pop   edx
                                <1> sysvideo_20_5:
00010696 31D2                   <1>       xor   edx, edx ; 0
                                <1>       ; edx = 0 -> invalid LFBINFO data
00010698 8B2D[60030300]         <1>       mov   ebp, [u.usp]  ; ebp points to user's registers
0001069E 895514                 <1>       mov   [ebp+20], edx ; return to user with EDX value
000106A1 E966D2FFFF             <1>       jmp   sysret
                                <1>
                                <1> sysvideo_21:
                                <1>       ; 04/01/2021
```

```
                                <1>       ; 03/12/2020
000106A6 80FF0A                 <1>       cmp    bh, 10
000106A9 0F82AA010000           <1>       jb     sysvideo_22 ; VESA VBE3 pmi parms
                                <1>       ; 23/12/2020
000106AF 0F845F020000           <1>       je     sysvideo_26 ; Video memory mapping
                                <1>
                                <1>       ; 04/01/2020
000106B5 80FF0B                 <1>       cmp    bh, 11
000106B8 0F87E1020000           <1>       ja     sysvideo_27
                                <1>
                                <1>       ; BH = 11
                                <1>       ; set/read DAC color registers (for 8bpp)
                                <1>
000106BE 80FB04                 <1>       cmp    bl, 4
000106C1 0F83AB000000           <1>       jnb    sysvideo_21_7; BMP file type palette
                                <1>                             ; handling
000106C7 F6C301                 <1>       test   bl, 1
000106CA 7555                   <1>       jnz    short sysvideo_21_4 ; set/write DAC colors
                                <1>
                                <1>       ; Read DAC color register or all DAC color registers
000106CC F6C302                 <1>       test   bl, 2  ; read single DAC color register
000106CF 7424                   <1>       jz     short sysvideo_21_2 ; read all DAC color regs
                                <1>
                                <1>       ; read single DAC color register
                                <1>       ; CL = DAC color register (index)
                                <1>
000106D1 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
000106D5 88C8                   <1>       mov    al, cl ; DAC color register
000106D7 31C9                   <1>       xor    ecx, ecx ; (this may not be necessary)
000106D9 EE                     <1>       out    dx, al
                                <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
000106DA B2C9                   <1>       mov    dl, 0C9h
000106DC EC                     <1>       in     al, dx
000106DD 88C4                   <1>       mov    ah, al ; red
000106DF EC                     <1>       in     al, dx
000106E0 88C1                   <1>       mov    cl, al ; green
000106E2 EC                     <1>       in     al, dx
000106E3 88C5                   <1>       mov    ch, al ; blue
000106E5 C1E108                 <1>       shl    ecx, 8
000106E8 88E1                   <1>       mov    cl, ah ; red
                                <1>       ; CL = Red, CH = Green, byte 3 = Blue, byte 4 = 0
                                <1> sysvideo_21_0:
000106EA 890D[64030300]         <1>       mov    [u.r0], ecx
                                <1> sysvideo_21_1:
000106F0 E917D2FFFF             <1>       jmp    sysret
                                <1> sysvideo_21_2:
                                <1>       ; read all DAC color registers
000106F5 89CB                   <1>       mov    ebx, ecx ; user's buffer address
000106F7 BF00600900             <1>       mov    edi, VBE3STACKADDR
000106FC 89FE                   <1>       mov    esi, edi
000106FE B900030000             <1>       mov    ecx, 768 ; 256*3
00010703 51                     <1>       push   ecx
00010704 66BAC703               <1>       mov    dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
00010708 28C0                   <1>       sub    al, al ; 0
0001070A EE                     <1>       out    dx, al
                                <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
0001070B B2C9                   <1>       mov    dl, 0C9h
                                <1> sysvideo_21_3:
0001070D EC                     <1>       in     al, dx
0001070E AA                     <1>       stosb
0001070F EC                     <1>       in     al, dx
00010710 AA                     <1>       stosb
00010711 EC                     <1>       in     al, dx
00010712 AA                     <1>       stosb
00010713 E2F8                   <1>       loop   sysvideo_21_3
00010715 59                     <1>       pop    ecx
                                <1>
00010716 89DF                   <1>       mov    edi, ebx ; user's buffer address
                                <1>       ;mov   esi, VBE3STACKADDR
                                <1>       ;mov   ecx, 256*3 = 768
00010718 E82A140000             <1>       call   transfer_to_user_buffer
0001071D 72D1                   <1>       jc     short sysvideo_21_1
                                <1>       ;mov   [u.r0], ecx ; actual transfer count
0001071F EBC9                   <1>       jmp    short sysvideo_21_0
                                <1>
                                <1> sysvideo_21_4:
                                <1>       ; Set/Write DAC color register or all registers
00010721 F6C302                 <1>       test   bl, 2 ; write/set single DAC color register
00010724 741C                   <1>       jz     short sysvideo_21_5 ; set all DAC color regs
                                <1>
                                <1>       ; set single DAC color register
                                <1>       ; CL = DAC color register (index)
                                <1>       ; (byte 1 = Red, byte 2 = Green, byte 3 = Blue)
                                <1>
00010726 66BAC803               <1>       mov    dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
0001072A 89C8                   <1>       mov    eax, ecx ; DAC color register (index)
0001072C C1E910                 <1>       shr    ecx, 16  ; CL = green, AH = Red
0001072F EE                     <1>       out    dx, al
                                <1>       ;mov   dx, 3C9h ; VGAREG_DAC_DATA
00010730 FEC2                   <1>       inc    dl
00010732 88E0                   <1>       mov    al, ah ; Red
00010734 EE                     <1>       out    dx, al
00010735 88C8                   <1>       mov    al, cl ; Green
00010737 EE                     <1>       out    dx, al
00010738 88E8                   <1>       mov    al, ch ; Blue
0001073A EE                     <1>       out    dx, al
                                <1>       ;rol   ecx, 8
0001073B C1E108                 <1>       shl    ecx, 8 ; 21/02/2021
0001073E 88E1                   <1>       mov    cl, ah ; Red
                                <1>              ; ecx = 00BBGGRRh
00010740 EBA8                   <1>       jmp    short sysvideo_21_0
                                <1>
                                <1> sysvideo_21_5:
                                <1>       ; write/set all DAC color registers
00010742 89CE                   <1>       mov    esi, ecx ; user's buffer address
```

```
9906 00010744 BF00600900          <1>       mov   edi, VBE3STACKADDR
9907 00010749 89FB                <1>       mov   ebx, edi
9908 0001074B B900030000          <1>       mov   ecx, 768 ; 256*3
9909 00010750 E83C140000          <1>       call  transfer_from_user_buffer
9910 00010755 7299                <1>       jc    short sysvideo_21_1
9911 00010757 890D[64030300]      <1>       mov   [u.r0], ecx ; actual transfer count
9912                              <1>
9913 0001075D 89DE                <1>       mov   esi, ebx ; VBE3STACKADDR
9914 0001075F 66BAC803            <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
9915 00010763 28C0                <1>       sub   al, al ; 0
9916 00010765 EE                  <1>       out   dx, al
9917                              <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
9918 00010766 FEC2                <1>       inc   dl
9919                              <1> sysvideo_21_6:
9920 00010768 AC                  <1>       lodsb
9921 00010769 EE                  <1>       out   dx, al
9922 0001076A AC                  <1>       lodsb
9923 0001076B EE                  <1>       out   dx, al
9924 0001076C AC                  <1>       lodsb
9925 0001076D EE                  <1>       out   dx, al
9926 0001076E E2F8                <1>       loop  sysvideo_21_6
9927 00010770 EB30                <1>       jmp   short sysvideo_21_9
9928                              <1>
9929                              <1> sysvideo_21_7:
9930                              <1>       ; BMP file type palette handling
9931                              <1>
9932 00010772 F6C301              <1>       test  bl, 1
9933 00010775 756E                <1>       jnz   short sysvideo_21_12 ; set/write DAC colors
9934                              <1>
9935                              <1>       ; Read DAC color register or all DAC color registers
9936 00010777 F6C302              <1>       test  bl, 2  ; read single DAC color register
9937 0001077A 742B                <1>       jz    short sysvideo_21_10 ; read all DAC color regs
9938                              <1>
9939                              <1>       ; read single DAC color register
9940                              <1>       ; CL = DAC color register (index)
9941                              <1>
9942 0001077C 66BAC703            <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9943 00010780 88C8                <1>       mov   al, cl ; DAC color register
9944 00010782 31C9                <1>       xor   ecx, ecx
9945 00010784 EE                  <1>       out   dx, al
9946                              <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
9947 00010785 B2C9                <1>       mov   dl, 0C9h
9948 00010787 EC                  <1>       in    al, dx
9949 00010788 C0E002              <1>       shl   al, 2
9950 0001078B 88C5                <1>       mov   ch, al ; red
9951 0001078D EC                  <1>       in    al, dx
9952 0001078E C0E002              <1>       shl   al, 2
9953 00010791 88C1                <1>       mov   cl, al ; green
9954 00010793 EC                  <1>       in    al, dx
9955 00010794 C0E002              <1>       shl   al, 2
9956                              <1>       ; 21/02/2021
9957 00010797 C1E108              <1>       shl   ecx, 8
9958 0001079A 88C1                <1>       mov   cl, al ; blue
9959                              <1>       ; CL = Blue, CH = Green, byte 3 = Red, byte 4 = 0
9960                              <1> sysvideo_21_8:
9961 0001079C 890D[64030300]      <1>       mov   [u.r0], ecx
9962                              <1> sysvideo_21_9:
9963 000107A2 E965D1FFFF          <1>       jmp   sysret
9964                              <1> sysvideo_21_10:
9965                              <1>       ; read all DAC color registers
9966 000107A7 89CD                <1>       mov   ebp, ecx ; user's buffer address
9967 000107A9 BF00600900          <1>       mov   edi, VBE3STACKADDR
9968 000107AE 89FE                <1>       mov   esi, edi
9969 000107B0 B900040000          <1>       mov   ecx, 1024 ; 256*4
9970 000107B5 51                  <1>       push  ecx
9971 000107B6 66BAC703            <1>       mov   dx, 3C7h ; VGAREG_DAC_READ_ADDRESS
9972 000107BA 28C0                <1>       sub   al, al ; 0
9973 000107BC EE                  <1>       out   dx, al
9974                              <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
9975 000107BD B2C9                <1>       mov   dl, 0C9h
9976                              <1> sysvideo_21_11:
9977 000107BF 31DB                <1>       xor   ebx, ebx
9978 000107C1 EC                  <1>       in    al, dx ; Red
9979 000107C2 C0E002              <1>       shl   al, 2
9980 000107C5 88C7                <1>       mov   bh, al
9981 000107C7 EC                  <1>       in    al, dx ; Green
9982 000107C8 C0E002              <1>       shl   al, 2
9983 000107CB 88C3                <1>       mov   bl, al
9984 000107CD EC                  <1>       in    al, dx ; Blue
9985 000107CE C0E002              <1>       shl   al, 2
9986 000107D1 C1E308              <1>       shl   ebx, 8
9987 000107D4 89D8                <1>       mov   eax, ebx ; 00RRGGBBh
9988 000107D6 AB                  <1>       stosd
9989 000107D7 E2E6                <1>       loop  sysvideo_21_11
9990 000107D9 59                  <1>       pop   ecx
9991                              <1>
9992 000107DA 89EF                <1>       mov   edi, ebp ; user's buffer address
9993                              <1>       ;mov  esi, VBE3STACKADDR
9994                              <1>       ;mov  ecx, 1024 = 4*256
9995 000107DC E866130000          <1>       call  transfer_to_user_buffer
9996 000107E1 72BF                <1>       jc    short sysvideo_21_9
9997                              <1>       ;mov  [u.r0], ecx ; actual transfer count
9998 000107E3 EBB7                <1>       jmp   short sysvideo_21_8
9999                              <1>
10000                             <1> sysvideo_21_12:
10001                             <1>       ; Set/Write DAC color register or all registers
10002 000107E5 F6C302             <1>       test  bl, 2 ; write/set single DAC color register
10003 000107E8 7427               <1>       jz    short sysvideo_21_13 ; set all DAC color regs
10004                             <1>
10005                             <1>       ; set single DAC color register
10006                             <1>       ; CL = DAC color register (index)
10007                             <1>       ; (byte 1 = Blue, byte 2 = Green, byte 3 = Red)
10008                             <1>
10009 000107EA 66BAC803           <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10010 000107EE 88C8               <1>       mov   al, cl ; DAC color register (index)
```

```
10011 000107F0 88EC              <1>       mov   ah, ch ; Blue
10012 000107F2 C1E910            <1>       shr   ecx, 16
10013 000107F5 EE                <1>       out   dx, al
10014                            <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
10015 000107F6 FEC2              <1>       inc   dl
10016 000107F8 88E8              <1>       mov   al, ch ; Red
10017 000107FA C0E802            <1>       shr   al, 2
10018 000107FD EE                <1>       out   dx, al
10019 000107FE 88C8              <1>       mov   al, cl ; Green
10020 00010800 C0E802            <1>       shr   al, 2
10021 00010803 EE                <1>       out   dx, al
10022 00010804 88E0              <1>       mov   al, ah ; Blue
10023 00010806 C0E802            <1>       shr   al, 2
10024 00010809 EE                <1>       out   dx, al
10025                            <1>       ;rol  ecx, 8
10026 0001080A C1E108            <1>       shl   ecx, 8 ; 21/02/2021
10027 0001080D 88E1              <1>       mov   cl, ah
10028 0001080F EB8B              <1>       jmp   short sysvideo_21_8
10029                            <1>
10030                            <1> sysvideo_21_13:
10031                            <1>       ; write/set all DAC color registers
10032 00010811 89CE              <1>       mov   esi, ecx ; user's buffer address
10033 00010813 BF00600900        <1>       mov   edi, VBE3STACKADDR
10034 00010818 89FB              <1>       mov   ebx, edi
10035 0001081A B900040000        <1>       mov   ecx, 1024 ; 256*4
10036 0001081F E86D130000        <1>       call  transfer_from_user_buffer
10037 00010824 722E              <1>       jc    short sysvideo_21_15
10038 00010826 890D[64030300]    <1>       mov   [u.r0], ecx ; actual transfer count
10039                            <1>
10040 0001082C 89DE              <1>       mov   esi, ebx ; VBE3STACKADDR
10041 0001082E 66BAC803          <1>       mov   dx, 3C8h ; VGAREG_DAC_WRITE_ADDRESS
10042 00010832 28C0              <1>       sub   al, al ; 0
10043 00010834 EE                <1>       out   dx, al
10044                            <1>       ;mov  dx, 3C9h ; VGAREG_DAC_DATA
10045 00010835 FEC2              <1>       inc   dl
10046                            <1> sysvideo_21_14:
10047 00010837 AD                <1>       lodsd
10048                            <1>       ; byte 0 = Blue, byte 1 = Green, byte 2 = Red
10049                            <1>       ; 21/02/2021
10050 00010838 89C3              <1>       mov   ebx, eax ; BL = Blue, BH = Green
10051 0001083A C1CB08            <1>       ror   ebx, 8 ; BL = Green, BH = Red
10052 0001083D 88F8              <1>       mov   al, bh
10053 0001083F C0E802            <1>       shr   al, 2
10054 00010842 EE                <1>       out   dx, al ; Red
10055 00010843 88D8              <1>       mov   al, bl
10056 00010845 C0E802            <1>       shr   al, 2
10057 00010848 EE                <1>       out   dx, al ; Green
10058 00010849 C1C308            <1>       rol   ebx, 8 ; BL = Blue
10059 0001084C 88D8              <1>       mov   al, bl
10060 0001084E C0E802            <1>       shr   al, 2
10061 00010851 EE                <1>       out   dx, al ; Blue
10062 00010852 E2E3              <1>       loop  sysvideo_21_14
10063                            <1> sysvideo_21_15:
10064 00010854 E9B3D0FFFF        <1>       jmp   sysret
10065                            <1>
10066                            <1> sysvideo_22:
10067                            <1>       ; 28/02/2021
10068                            <1>       ; 22/01/2021
10069                            <1>       ; 17/01/2021
10070                            <1>       ; 04/12/2020
10071                            <1>       ; 03/12/2020
10072                            <1>       ; BH = 9
10073                            <1>       ; Set/Get VESA VBE3 protected mode interface params
10074                            <1>
10075                            <1>       ; 22/01/2021
10076                            <1>       ;cmp  byte [vbe3], 3
10077                            <1>       ;jne  short sysvideo_25 ; not applicable if
10078                            <1>                               ; vbe3 compatible video bios
10079                            <1>                               ; is not detected by kernel
10080 00010859 80FB02            <1>       cmp   bl, 2
10081                            <1>       ;ja   short sysvideo_25 ; bl > 2 not implemented
10082                            <1>       ; 17/01/2021
10083 0001085C 7716              <1>       ja    short sysvideo_22_0 ; srvs flag sub function
10084                            <1>       ;jb   short sysvideo_23
10085                            <1>
10086                            <1>       ; 21/01/2021
10087 0001085E 803D[5C090000]03  <1>       cmp   byte [vbe3], 3
10088                            <1>       ;jne  short sysvideo_25 ; not applicable if
10089                            <1>                               ; vbe3 compatible video bios
10090                            <1>                               ; is not detected by kernel
10091 00010865 75ED              <1>       jne   short sysvideo_21_15 ; 28/02/2021
10092                            <1>
10093 00010867 80FB01            <1>       cmp   bl, 1
10094 0001086A 7673              <1>       jna   short sysvideo_23
10095                            <1>
10096 0001086C 8A1D[1C120300]    <1>       mov   bl, [pmi32] ; Video bios 32 bit PMI functions
10097 00010872 EB78              <1>       jmp   short sysvideo_24
10098                            <1>
10099                            <1> sysvideo_22_0:
10100                            <1>       ; 17/01/2021
10101                            <1>       ; save/restore video state user permission
10102 00010874 80FB05            <1>       cmp   bl, 5
10103 00010877 771E              <1>       ja    short sysvideo_22_2
10104 00010879 7208              <1>       jb    short sysvideo_22_1
10105                            <1>       ; get srvs flag value/status
10106 0001087B 8A1D[80120300]    <1>       mov   bl, [srvsf] ; 0 = disabled, 1 = enabled
10107 00010881 EB2C              <1>       jmp   short sysvideo_22_3
10108                            <1>
10109                            <1> sysvideo_22_1:
10110                            <1>       ; permission (root and multi tasking) check
10111 00010883 E836000000        <1>       call  sysvideo_22_4
10112 00010888 736A              <1>       jnc   short sysvideo_25 ; not permitted !
10113                            <1>       ; cf = 1
10114 0001088A 80EB03            <1>       sub   bl, 3 ; disable = 0, enable = 1
10115                            <1>       ; 22/01/2021
```

```
10116 0001088D 881D[80120300]      <1>       mov    [srvsf], bl
10117 00010893 FEC3                <1>       inc    bl ; 1 = disabled, 2 = enabled
10118 00010895 EB18                <1>       jmp    short sysvideo_22_3
10119                              <1>
10120                              <1> sysvideo_22_2:
10121 00010897 80FB06              <1>       cmp    bl, 6
10122                              <1>       ;ja    short sysvideo_25 ; invalid/unimplemented
10123                              <1>       ; 28/02/2021
10124 0001089A 7733                <1>       ja     short sysvideo_22_6
10125                              <1>       ; get VESA VBE number/status
10126 0001089C 8A25[5C090000]      <1>       mov    ah, [vbe3] ; vbe3 = 3, vbe2 = 2, others = 0
10127 000108A2 A0[5D090000]        <1>       mov    al, [vbe2bios] ; bochs/qemu/vbox emulator status
10128 000108A7 66A3[64030300]      <1>       mov    [u.r0], ax
10129 000108AD EB45                <1>       jmp    short sysvideo_25
10130                              <1>
10131                              <1> sysvideo_22_3:
10132                              <1>       ; 22/01/2021
10133 000108AF 8A3D[81120300]      <1>       mov    bh, [srvso] ; state options (> 80h -> svga)
10134 000108B5 66891D[64030300]    <1>       mov    [u.r0], bx ; function result is return value
10135 000108BC EB36                <1>       jmp    short sysvideo_25
10136                              <1>
10137                              <1> sysvideo_22_4:
10138                              <1>       ; 17/01/2021 - permission will be given by root only
10139 000108BE 803D[AA960100]00    <1>       cmp    byte [multi_tasking], 0 ; in single user mode
10140 000108C5 7707                <1>       ja     short sysvideo_22_5
10141                              <1>       ; 19/01/2021
10142 000108C7 803D[B0030300]01    <1>       cmp    byte [u.uid], 1 ; ([u.uid] = 0 -> root)
10143                              <1> sysvideo_22_5:
10144                              <1>       ; [multi_tasking] = 0 & [u.uid] = 0 -> CF = 1
10145                              <1>       ; otherwise -> CF = 0
10146 000108CE C3                 <1>       retn
10147                              <1>
10148                              <1> sysvideo_22_6:
10149                              <1>       ; 28/02/2021
10150 000108CF 80FB09              <1>       cmp    bl, 9
10151 000108D2 7720                <1>       ja     short sysvideo_25 ; invalid/unimplemented
10152 000108D4 7436                <1>       je     short sysvideo_22_9
10153 000108D6 80FB08              <1>       cmp    bl, 8
10154 000108D9 721E                <1>       jb     short sysvideo_22_7
10155                              <1>
10156                              <1>       ; BL = 8
10157                              <1>       ; Set default true color bpp to 24
10158                              <1>
10159 000108DB B318                <1>       mov    bl, 24
10160                              <1>       ;mov   [truecolor], al     ; 24bpp (RRGGBBh)
10161                              <1>       ;mov   [u.r0], al
10162                              <1>       ;jmp   short sysvideo_25
10163 000108DD EB25                <1>       jmp    short sysvideo_22_8
10164                              <1>
10165                              <1> sysvideo_23:
10166                              <1>       ; 17/01/2021
10167                              <1>       ; permission (root and multi tasking) check
10168 000108DF E8DAFFFFFF          <1>       call   sysvideo_22_4
10169 000108E4 730E                <1>       jnc    short sysvideo_25 ; not permitted !
10170                              <1>
10171 000108E6 881D[1C120300]      <1>       mov    [pmi32], bl ; 1 = enabled, 0 = disabled
10172                              <1> sysvideo_24:
10173 000108EC FEC3                <1>       inc    bl
10174                              <1> sysvideo_22_10:    ; 28/02/2021
10175 000108EE 881D[64030300]      <1>       mov    [u.r0], bl ; function result is return value
10176                              <1> sysvideo_25:
10177 000108F4 E913D0FFFF          <1>       jmp    sysret
10178                              <1>
10179                              <1> sysvideo_22_7:
10180                              <1>       ; BL = 7
10181                              <1>       ; Set default true color bpp to 32
10182                              <1>       ; (it will set if [VBE3]=3)
10183                              <1>
10184                              <1>       ; Note: This sub function is used to set 24bpp
10185                              <1>       ; VESA VBE video modes to 32bpp.. because,
10186                              <1>       ; old hardware uses 24 bpp but new video hardware
10187                              <1>       ; uses 32bpp for same VESA VBE truecolor modes.
10188                              <1>       ; (For example: VBE mode 112h is 640*480, 24bpp but
10189                              <1>       ; new hardware uses/apply it as 640*480, 32bpp.)
10190                              <1>       ; So, TRDOS 386 v2.0.3 kernel will check [truecolor]
10191                              <1>       ; status is 32 bpp or not and it will change 24bpp
10192                              <1>       ; to 32bpp if default [truecolor] value is 32, for
10193                              <1>       ; same video mode number.
10194                              <1>
10195 000108F9 803D[5C090000]03    <1>       cmp    byte [vbe3], 3
10196 00010900 75F2                <1>       jne    short sysvideo_25 ; Only applicable
10197                              <1>                                ; for VBE3 video hardware!
10198 00010902 B320                <1>       mov    bl, 32
10199                              <1> sysvideo_22_8:
10200 00010904 881D[DB860100]      <1>       mov    [truecolor], bl     ; 32bpp (00RRGGBBh)
10201                              <1>       ;mov   [u.r0], bl
10202                              <1>       ;jmp   short sysvideo_25
10203 0001090A EBE2                <1>       jmp    short sysvideo_22_10
10204                              <1>
10205                              <1> sysvideo_22_9:
10206                              <1>       ; BL = 9
10207                              <1>       ; Return default true color bpp
10208 0001090C 8A1D[DB860100]      <1>       mov    bl, [truecolor]
10209 00010912 EBDA                <1>       jmp    short sysvideo_22_10
10210                              <1> ;sysvideo_22_10:
10211                              <1>       ;mov   [u.r0], bl
10212                              <1>       ;jmp   sysret
10213                              <1>
10214                              <1> sysvideo_26:
10215                              <1>       ; 23/12/2020
10216                              <1>       ; BH = 10
10217                              <1>       ; Map video memory to user's buffer
10218                              <1>       ; (multiuser/owner r/w permisions are ignored
10219                              <1>       ;  for current TRDOS 386 version !)
10220                              <1>
```

```
10221 00010914 6681E100F0          <1>       and    cx, ~4095  ; clear low 12 bits
10222 00010919 09C9                <1>       or     ecx, ecx ; start address of user's buffer
10223 0001091B 74D7                <1>       jz     short sysvideo_25 ; error !
10224                              <1>
10225 0001091D 80FB01              <1>       cmp    bl, 1  ; VGA memory mapping ?
10226 00010920 740E                <1>       je     short sysvideo_26_1
10227 00010922 7718                <1>       ja     short sysvideo_26_2
10228                              <1> sysvideo_26_0:
10229                              <1>       ; BL = 0 : CGA memory (0B8000h) map (32K)
10230 00010924 B800800B00          <1>       mov    eax, 0B8000h
10231 00010929 BB00800000          <1>       mov    ebx, 32768
10232 0001092E EB37                <1>       jmp    short sysvideo_26_3
10233                              <1> sysvideo_26_1:
10234                              <1>       ; BL = 1 : VGA memory (0A0000h) map (64K)
10235 00010930 B800000A00          <1>       mov    eax, 0A0000h
10236 00010935 BB00000100          <1>       mov    ebx, 65536
10237 0001093A EB2B                <1>       jmp    short sysvideo_26_3
10238                              <1> sysvideo_26_2:
10239                              <1>       ; BL = 2 : SVGA memory (LFB) map to user's buffer
10240 0001093C 803D[5C090000]02    <1>       cmp    byte [vbe3], 2  ; VESA VBE 2/3 vbios ready ?
10241 00010943 72AF                <1>       jb     short sysvideo_25  ; no, error !
10242 00010945 6681E200F0          <1>       and    dx, ~4095 ; clear low 12 bits
10243 0001094A 09D2                <1>       or     edx, edx  ; buffer size in bytes
10244 0001094C 74A6                <1>       jz     short sysvideo_25  ; error
10245 0001094E 89D3                <1>       mov    ebx, edx
10246 00010950 A1[2C120300]        <1>       mov    eax, [LFB_ADDR] ; [LFB_Info+LFBINFO.LFB_addr]
10247 00010955 21C0                <1>       and    eax, eax
10248 00010957 7425                <1>       jz     short sysvideo_26_5
10249                              <1>                              ; (LFB parms are not set yet)
10250 00010959 3B1D[30120300]      <1>       cmp    ebx, [LFB_SIZE] ; [LFB_Info+LFBINFO.LFB_size]
10251 0001095F 7606                <1>       jna    short sysvideo_26_3
10252 00010961 8B1D[30120300]      <1>       mov    ebx, [LFB_SIZE]
10253                              <1> sysvideo_26_3:
10254 00010967 52                  <1>       push   edx
10255 00010968 53                  <1>       push   ebx    ; buffer size in bytes
10256 00010969 51                  <1>       push   ecx    ; user's buffer address
10257 0001096A 87D9                <1>       xchg   ebx, ecx
10258 0001096C C1E90C              <1>       shr    ecx, 12 ; convert buffer size to page count
10259 0001096F E88E5DFFFF          <1>       call   direct_memory_access
10260 00010974 59                  <1>       pop    ecx    ; user's buffer address
10261 00010975 5B                  <1>       pop    ebx    ; buffer size
10262 00010976 5A                  <1>       pop    edx
10263                              <1>       ;jc    short sysvideo_25 ; error !
10264                              <1>                              ; [u.r0] = 0
10265                              <1>       ; 28/02/2021
10266 00010977 7234                <1>       jc     short sysvideo_27_0 ; error !
10267                              <1>
10268                              <1> ;sysvideo_26_4:
10269                              <1>       ;mov   ebp, [u.usp] ; ebp points to user's registers
10270                              <1>       ;mov   [ebp+20], edx ; return to user with EDX value
10271                              <1>       ;mov   [ebp+16], ebx ; EBX
10272                              <1>       ;mov   [ebp+24], ecx ; ECX
10273                              <1>       ; eax = physical address of video memory (LFB)
10274                              <1>       ;mov   [u.r0], eax
10275                              <1>       ;jmp   sysret
10276 00010979 E919FCFFFF          <1>       jmp    sysvideo_26_4
10277                              <1>
10278                              <1> sysvideo_26_5:
10279 0001097E 66A1[EB0E0000]      <1>       mov    ax, [def_LFB_addr] ; default LFB for mode 118h
10280                              <1>       ; ah must be 0C0h or 0D0h or E0h
10281                              <1>       ; others are nonsence !?
10282 00010984 08E4                <1>       or     ah, ah
10283                              <1>       ;jz    short sysvideo_25 ; invalid lfb addr or
10284                              <1>                              ; it is not a vbe2 -bochs emu-
10285                              <1>                              ; or vbe3 -real- video bios
10286                              <1>       ; 28/02/2021
10287 00010986 7425                <1>       jz     short sysvideo_27_0 ; invalid LFB address
10288                              <1>
10289 00010988 80FCF0              <1>       cmp    ah, 0F0h
10290                              <1>       ;jnb   short sysvideo_25 ; nonsence !?
10291                              <1>       ; 28/02/2021
10292 0001098B 7320                <1>       jnb    short sysvideo_27_0 ; nonsence !?
10293                              <1>
10294 0001098D C1E010              <1>       shl    eax, 16
10295                              <1>       ;jz    short sysvideo_25 ; eax = 0
10296                              <1>
10297 00010990 81FB00907E00        <1>       cmp    ebx, 1920*1080*4 ; maximum value of possible
10298                              <1>                               ; buffer sizes
10299 00010996 76CF                <1>       jna    short sysvideo_26_3  ; buffer size is proper
10300                              <1>       ; resize buffer to fit 4GB limit
10301 00010998 BB00907E00          <1>       mov    ebx, 1920*1080*4
10302 0001099D EBC8                <1>       jmp    short sysvideo_26_3
10303                              <1>
10304                              <1> sysvideo_27:
10305                              <1>       ; 16/02/2021
10306                              <1>       ; 18/01/2021
10307 0001099F 80FF0C              <1>       cmp    bh, 12
10308 000109A2 0F8778010000        <1>       ja     sysvideo_28 ; 19/01/2021
10309                              <1>
10310                              <1>       ; BH = 12
10311                              <1>       ; Font sub functions.
10312                              <1>       ; 12/02/2021
10313                              <1>       ; 11/01/2021
10314                              <1>       ; 10/01/2021
10315                              <1>       ;   BL = 0 : Disable system font overwrite
10316                              <1>       ;   BL = 1 : Enable system font overwrite
10317                              <1>       ;   BL = 2 : Read system font 8x8
10318                              <1>       ;   BL = 3 : Read system font 8x14
10319                              <1>       ;   BL = 4 : Read system font 8x16
10320                              <1>       ;   BL = 5 : Read user defined font 8x8
10321                              <1>       ;   BL = 6 : Read user defined font 8x16
10322                              <1>       ;   BL = 7 : Write system font 8x8
10323                              <1>       ;   BL = 8 : Write system font 8x14
10324                              <1>       ;   BL = 9 : Write system font 8x16
10325                              <1>       ;   BL = 10 : Write user defined font 8x8
```

```
10326                                  <1>       ;   BL = 11 : Write user defined font 8x16
10327                                  <1>       ;
10328                                  <1>       ;  BL > 11 : invalid (not implemented)
10329                                  <1>       ;
10330                                  <1>       ; For BL = 1 to 11
10331                                  <1>       ;   ECX = number of characters (<= 256)
10332                                  <1>       ;   EDX = first character (ascii code in DL)
10333                                  <1>       ;   ESI = user's buffer address
10334                                  <1>       ;
10335                                  <1>       ; Return: EAX = character count
10336                                  <1>
10337 000109A8 80FB0B                 <1>       cmp    bl, 11
10338 000109AB 7605                   <1>       jna    short sysvideo_27_1
10339                                  <1> sysvideo_27_0:
10340 000109AD E95ACFFFFF             <1>       jmp    sysret ; not implemented yet !
10341                                  <1> sysvideo_27_1:
10342 000109B2 66B80001               <1>       mov    ax, 256
10343 000109B6 08DB                   <1>       or     bl, bl
10344 000109B8 750E                   <1>       jnz    short sysvideo_27_3
10345                                  <1>
10346                                  <1>       ; bl = 0
10347                                  <1>       ; disable system font overwrite
10348                                  <1>
10349 000109BA 8025[7E120300]7F       <1>       and    byte [ufont], 7Fh ; clear bit 7
10350                                  <1> sysvideo_27_2:
10351                                  <1>       ;mov   word [u.r0], 256  ; > 0 -> successful
10352 000109C1 A3[64030300]           <1>       mov    [u.r0], eax ; 256
10353 000109C6 EBE5                   <1>       jmp    short sysvideo_27_0
10354                                  <1> sysvideo_27_3:
10355 000109C8 80FB01                 <1>       cmp    bl, 1
10356 000109CB 7710                   <1>       ja     short sysvideo_27_4
10357                                  <1>
10358                                  <1>       ; bl = 1
10359                                  <1>       ; enable system font overwrite
10360                                  <1>       ;      if [multi_tasking]= 0 and [u.uid] = 0
10361                                  <1>
10362                                  <1>       ;cmp   byte [multi_tasking], 0
10363                                  <1>       ;                    ; multi tasking enabled ?
10364                                  <1>       ;ja    short sysvideo_27_0 ; yes
10365                                  <1>       ;; 19/01/2021
10366                                  <1>       ;; system maintenance or single user mode
10367                                  <1>       ;cmp   byte [u.uid], 0  ; root ?
10368                                  <1>       ;ja    short sysvideo_27_0 ; no
10369                                  <1>
10370                                  <1>       ; 19/01/2021
10371                                  <1>       ; multi tasking & root check
10372 000109CD E8ECFEFFFF             <1>       call   sysvideo_22_4
10373 000109D2 73D9                   <1>       jnc    short sysvideo_27_0 ; not permitted
10374                                  <1>
10375                                  <1>       ; [multi_tasking]= 0 and [u.uid] = 0
10376                                  <1>
10377 000109D4 800D[7E120300]80       <1>       or     byte [ufont], 80h ; set bit 7
10378                                  <1>
10379 000109DB EBE4                   <1>       jmp    short sysvideo_27_2
10380                                  <1>
10381                                  <1> sysvideo_27_4:
10382 000109DD 09C9                   <1>       or     ecx, ecx
10383 000109DF 74CC                   <1>       jz     short sysvideo_27_0
10384 000109E1 21D2                   <1>       and    edx, edx
10385 000109E3 7410                   <1>       jz     short sysvideo_27_4_0
10386                                  <1>       ;mov   ax, 256
10387 000109E5 39C1                   <1>       cmp    ecx, eax ; 256
10388 000109E7 77C4                   <1>       ja     short sysvideo_27_0
10389 000109E9 48                     <1>       dec    eax
10390 000109EA 39C2                   <1>       cmp    edx, eax ; 255
10391 000109EC 77BF                   <1>       ja     short sysvideo_27_0
10392 000109EE 40                     <1>       inc    eax
10393 000109EF 29D0                   <1>       sub    eax, edx ; 256 - DX
10394 000109F1 39C8                   <1>       cmp    eax, ecx
10395 000109F3 72B8                   <1>       jb     short sysvideo_27_0
10396                                  <1>
10397                                  <1> sysvideo_27_4_0:
10398 000109F5 89F5                   <1>       mov    ebp, esi
10399                                  <1>
10400 000109F7 80FB06                 <1>       cmp    bl, 6
10401 000109FA 776C                   <1>       ja     short sysvideo_27_13
10402 000109FC 7210                   <1>       jb     short sysvideo_27_5
10403                                  <1>       ; bl = 6
10404 000109FE F605[7E120300]10       <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
10405 00010A05 74A6                   <1>       jz     short sysvideo_27_0
10406                                  <1>       ; read 8x16 user defined font
10407 00010A07 BE00400900             <1>       mov    esi, VGAFONT16USER
10408 00010A0C EB0C                   <1>       jmp    short sysvideo_27_6
10409                                  <1> sysvideo_27_5:
10410 00010A0E 80FB04                 <1>       cmp    bl, 4
10411 00010A11 723D                   <1>       jb     short sysvideo_27_11
10412 00010A13 7723                   <1>       ja     short sysvideo_27_9
10413                                  <1>       ; bl = 4
10414                                  <1>       ; read 8x16 system font
10415 00010A15 BE[A8760100]           <1>       mov    esi, vgafont16
10416                                  <1> sysvideo_27_6:
10417                                  <1>       ; read 8x16 font
10418 00010A1A 66C1E204               <1>       shl    dx, 4 ; * 16
10419 00010A1E 66C1E104               <1>       shl    cx, 4 ; * 16 ; 16 bytes per char
10420                                  <1> sysvideo_27_7:
10421 00010A22 89EF                   <1>       mov    edi, ebp
10422                                  <1>       ;add   edi, edx ; 16/02/2021
10423 00010A24 01D6                   <1>       add    esi, edx
10424                                  <1>       ; ecx = byte count
10425                                  <1>       ; esi = source (in system memory)
10426                                  <1>       ; edi = destination (in user memory)
10427 00010A26 E81C110000             <1>       call   transfer_to_user_buffer
10428 00010A2B 7206                   <1>       jc     short sysvideo_27_8
10429 00010A2D 890D[64030300]         <1>       mov    [u.r0], ecx
10430                                  <1> sysvideo_27_8:
```

```
10431 00010A33 E9D4CEFFFF          <1>       jmp    sysret
10432                              <1> sysvideo_27_9:
10433                              <1>       ; bl = 5
10434 00010A38 F605[7E120300]08    <1>       test   byte [ufont], 8 ; 8x8 user font loaded ?
10435 00010A3F 74F2                <1>       jz     short sysvideo_27_8
10436                              <1>       ; read 8x8 user defined font
10437 00010A41 BE00500900          <1>       mov    esi, VGAFONT8USER
10438                              <1> sysvideo_27_10:
10439                              <1>       ; read 8x8 font
10440 00010A46 66C1E203            <1>       shl    dx, 3 ; * 8
10441 00010A4A 66C1E103            <1>       shl    cx, 3 ; * 8 ; 8 bytes per char
10442 00010A4E EBD2                <1>       jmp    short sysvideo_27_7
10443                              <1>
10444                              <1> sysvideo_27_11:
10445 00010A50 80FB03              <1>       cmp    bl, 3  ; 8x14 system font
10446 00010A53 720C                <1>       jb     short sysvideo_27_12 ; 8x8 system font
10447                              <1>       ; bl = 3
10448                              <1>       ; read 8x14 system font
10449                              <1>       ;mov   al, 14
10450                              <1>       ;mul   dl
10451                              <1>       ;mov   dx, ax
10452                              <1>       ;push  edx
10453                              <1>       ;mov   ax, 14
10454                              <1>       ;mul   cx
10455                              <1>       ;mov   cx, ax
10456                              <1>       ;pop   edx
10457 00010A55 E8A9000000          <1>       call   sysvideo_27_14
10458 00010A5A BE[A8680100]        <1>       mov    esi, vgafont14
10459 00010A5F EBC1                <1>       jmp    short sysvideo_27_7
10460                              <1>
10461                              <1> sysvideo_27_12:
10462                              <1>       ; bl = 2
10463                              <1>       ; read 8x8 system font
10464 00010A61 BE[A8600100]        <1>       mov    esi, vgafont8
10465 00010A66 EBDE                <1>       jmp    short sysvideo_27_10
10466                              <1>
10467                              <1> sysvideo_27_13:
10468                              <1>       ; overwrite font
10469 00010A68 80FB0A              <1>       cmp    bl, 10
10470 00010A6B 7772                <1>       ja     short sysvideo_27_22 ; 8x16 user font
10471 00010A6D 7224                <1>       jb     short sysvideo_27_15
10472                              <1>       ; bl = 10
10473 00010A6F BF00500900          <1>       mov    edi, VGAFONT8USER
10474 00010A74 F605[7E120300]08    <1>       test   byte [ufont], 8 ; 8x8 user font loaded ?
10475 00010A7B 7558                <1>       jnz    short sysvideo_27_21 ; yes
10476 00010A7D 08ED                <1>       or     ch, ch ; cx = 256
10477                              <1>       ;jnz   short sysvideo_27_21 ; 256 chars
10478 00010A7F 7406                <1>       jz     short sysvideo_27_13_0
10479 00010A81 66B90008            <1>       mov    cx, 8*256
10480 00010A85 EB37                <1>       jmp    short sysvideo_27_18_0
10481                              <1> sysvideo_27_13_0:
10482                              <1>       ; copy system font to user font before overwrite
10483 00010A87 BE[A8600100]        <1>       mov    esi, vgafont8
10484                              <1>       ;push  edi
10485                              <1>       ;push  ecx
10486                              <1>       ;mov   cl, 64
10487                              <1>       ;rep   movsd
10488                              <1>       ;pop   ecx
10489                              <1>       ;pop   edi
10490                              <1>       ;mov   esi, ebp ; user's font buffer
10491 00010A8C E884000000          <1>       call   sysvideo_27_23
10492 00010A91 EB42                <1>       jmp    short sysvideo_27_21
10493                              <1>
10494                              <1> sysvideo_27_15:
10495                              <1>       ; check system font overwrite permission
10496 00010A93 F605[7E120300]80    <1>       test   byte [ufont], 80h
10497 00010A9A 7497                <1>       jz     short sysvideo_27_8
10498                              <1>
10499 00010A9C 80FB08              <1>       cmp    bl, 8
10500 00010A9F 773E                <1>       ja     short sysvideo_27_22 ; 8x16 system font
10501 00010AA1 722D                <1>       jb     short sysvideo_27_20 ; 8x8 system font
10502                              <1>       ; bl = 8
10503                              <1>       ; overwrite 8x14 system font
10504                              <1>       ;mov   al, 14
10505                              <1>       ;mul   dl
10506                              <1>       ;mov   dx, ax
10507                              <1>       ;push  edx
10508                              <1>       ;mov   ax, 14
10509                              <1>       ;mul   cx
10510                              <1>       ;mov   cx, ax
10511                              <1>       ;pop   edx
10512 00010AA3 E85B000000          <1>       call   sysvideo_27_14
10513 00010AA8 BF[A8680100]        <1>       mov    edi, vgafont14
10514 00010AAD EB0D                <1>       jmp    short sysvideo_27_18
10515                              <1> sysvideo_27_16:
10516                              <1>       ; bl = 9
10517                              <1>       ; overwrite 8x16 system font
10518 00010AAF BF[A8760100]        <1>       mov    edi, vgafont16
10519                              <1> sysvideo_27_17:
10520                              <1>       ; overwrite 8x16 font
10521 00010AB4 66C1E204            <1>       shl    dx, 4 ; * 16
10522 00010AB8 66C1E104            <1>       shl    cx, 4 ; * 16 ; 16 bytes per char
10523                              <1> sysvideo_27_18:
10524 00010ABC 01D7                <1>       add    edi, edx
10525                              <1>       ;add   esi, edx ; 16/02/2021
10526                              <1> sysvideo_27_18_0:
10527                              <1>       ; ecx = byte count
10528                              <1>       ; esi = source (in user memory)
10529                              <1>       ; edi = destination (in system memory)
10530 00010ABE E8CE100000          <1>       call   transfer_from_user_buffer
10531 00010AC3 7206                <1>       jc     short sysvideo_27_19
10532 00010AC5 890D[64030300]      <1>       mov    [u.r0], ecx
10533                              <1> sysvideo_27_19:
10534 00010ACB E93CCEFFFF          <1>       jmp    sysret
10535                              <1> sysvideo_27_20:
```

```
10536                                   <1>       ; bl = 7
10537                                   <1>       ; overwrite 8x8 system font
10538 00010AD0 BF[A8600100]             <1>       mov    edi, vgafont8
10539                                   <1> sysvideo_27_21:
10540                                   <1>       ; write 8x8 font
10541 00010AD5 66C1E203                 <1>       shl    dx, 3 ; * 8
10542 00010AD9 66C1E103                 <1>       shl    cx, 3 ; * 8 ; 8 bytes per char
10543 00010ADD EBDD                     <1>       jmp    short sysvideo_27_18
10544                                   <1> sysvideo_27_22:
10545                                   <1>       ; bl = 11
10546                                   <1>       ; overwrite 8x16 user defined font
10547 00010ADF BF00400900               <1>       mov    edi, VGAFONT16USER
10548 00010AE4 F605[7E120300]10         <1>       test   byte [ufont], 16 ; 8x16 user font loaded ?
10549 00010AEB 75C7                     <1>       jnz    short sysvideo_27_17 ; yes
10550 00010AED 08ED                     <1>       or     ch, ch ; cx = 256
10551                                   <1>       ;jnz   short sysvideo_27_17 ; 256 chars
10552 00010AEF 7406                     <1>       jz     short sysvideo_27_22_0
10553 00010AF1 66B90010                 <1>       mov    cx, 16*256
10554 00010AF5 EBC7                     <1>       jmp    short sysvideo_27_18_0
10555                                   <1> sysvideo_27_22_0:
10556                                   <1>       ; copy system font to user font before overwrite
10557 00010AF7 BE[A8760100]             <1>       mov    esi, vgafont16
10558                                   <1>       ;push  edi
10559                                   <1>       ;push  ecx
10560                                   <1>       ;mov   cl, 64
10561                                   <1>       ;rep   movsd
10562                                   <1>       ;pop   ecx
10563                                   <1>       ;pop   edi
10564                                   <1>       ;mov   esi, ebp ; user's font buffer
10565 00010AFC E814000000               <1>       call   sysvideo_27_23
10566 00010B01 EBB1                     <1>       jmp    short sysvideo_27_17
10567                                   <1>
10568                                   <1> sysvideo_27_14:
10569                                   <1>       ; 16/02/2021
10570 00010B03 52                       <1>       push   edx
10571 00010B04 66B80E00                 <1>       mov    ax, 14
10572 00010B08 66F7E1                   <1>       mul    cx
10573 00010B0B 89C1                     <1>       mov    ecx, eax
10574 00010B0D 5A                       <1>       pop    edx
10575 00010B0E B00E                     <1>       mov    al, 14
10576 00010B10 F6E2                     <1>       mul    dl
10577 00010B12 89C2                     <1>       mov    edx, eax
10578 00010B14 C3                       <1>       retn
10579                                   <1>
10580                                   <1>       ;mov   al, 14
10581                                   <1>       ;mul   dl
10582                                   <1>       ;mov   dx, ax
10583                                   <1>       ;push  edx
10584                                   <1>       ;; 12/02/2021
10585                                   <1>       ;mov   ax, 14
10586                                   <1>       ;;mov  eax, 14
10587                                   <1>       ;;mul  cx
10588                                   <1>       ;mul   ecx
10589                                   <1>       ;;mov  cx, ax
10590                                   <1>       ;mov   ecx, eax
10591                                   <1>       ;pop   edx
10592                                   <1>       ;retn
10593                                   <1>
10594                                   <1> sysvideo_27_23:
10595 00010B15 57                       <1>       push   edi
10596 00010B16 51                       <1>       push   ecx
10597 00010B17 B140                     <1>       mov    cl, 64
10598 00010B19 F3A5                     <1>       rep    movsd
10599 00010B1B 59                       <1>       pop    ecx
10600 00010B1C 5F                       <1>       pop    edi
10601 00010B1D 89EE                     <1>       mov    esi, ebp ; user's font buffer
10602 00010B1F C3                       <1>       retn
10603                                   <1>
10604                                   <1> sysvideo_28:
10605                                   <1>       ; 24/01/2021
10606                                   <1>       ; 23/01/2021
10607                                   <1>       ; 18/01/2021
10608 00010B20 80FF0E                   <1>       cmp    bh, 14
10609 00010B23 0F8275010000             <1>       jb     sysvideo_29
10610 00010B29 0F8754020000             <1>       ja     sysvideo_30
10611                                   <1>
10612                                   <1>       ; BH = 14
10613                                   <1>       ; Save/Restore Super VGA video state
10614                                   <1>
10615                                   <1>       ; BL = options
10616                                   <1>       ;      bit 0 - Save (0) or Restore (1)
10617                                   <1>       ;      bit 1 - controller hardware state
10618                                   <1>       ;      bit 2 - BIOS data state
10619                                   <1>       ;      bit 3 - DAC state
10620                                   <1>       ;      bit 4 - (extended) Register state
10621                                   <1>       ;      bit 5 - system (0) or user (1) memory
10622                                   <1>       ;      bit 6 - verify without transfer
10623                                   <1>       ;      bit 7 - not used (must be 0)
10624                                   <1>
10625                                   <1>       ; ECX = Buffer address or VideoStateID
10626                                   <1>
10627 00010B2F 803D[5C090000]02         <1>       cmp    byte [vbe3], 2 ;  VESA VBE2 or VBE3 ?
10628 00010B36 7717                     <1>       ja     short sysvideo_28_0 ; yes
10629 00010B38 7210                     <1>       jb     short sysvideo_28_16 ; not a SVGA sys !
10630                                   <1>
10631                                   <1>       ; == VBE2 ==
10632                                   <1>       ; Check Bochs/Qemu/VirtualBox PC emulator
10633                                   <1>       ; (vbe2 is usable only for emulator's vbios)
10634 00010B3A 8A25[5D090000]           <1>       mov    ah, [vbe2bios]
10635 00010B40 80FCC0                   <1>       cmp    ah, 0C0h
10636 00010B43 7205                     <1>       jb     short sysvideo_28_16 ; unknown vbios !
10637 00010B45 80FCC5                   <1>       cmp    ah, 0C5h
10638 00010B48 7605                     <1>       jna    short sysvideo_28_0
10639                                   <1>              ; Use kernel's vbios functions (video.s)
10640                                   <1> sysvideo_28_16:
```

```
10641                                  <1>       ; unknown vbios !
10642 00010B4A E9BDCDFFFF              <1>       jmp    sysret
10643                                  <1>
10644                                  <1> sysvideo_28_0:
10645 00010B4F 80FB7F                  <1>       cmp    bl, 7Fh
10646 00010B52 77F6                    <1>       ja     short sysvideo_28_16 ; unknown options
10647                                  <1>
10648 00010B54 88DA                    <1>       mov    dl, bl
10649 00010B56 80E21F                  <1>       and    dl, 1Fh
10650 00010B59 D0EA                    <1>       shr    dl, 1
10651 00010B5B 74ED                    <1>       jz     short sysvideo_28_16 ; invalid !
10652                                  <1>       ; DL = VBE Function 4F04h Save/Restore options
10653                                  <1>       ;  bit 0 : controller hardware state
10654                                  <1>       ;  bit 1 : BIOS data state
10655                                  <1>       ;  bit 2 : DAC state
10656                                  <1>       ;  bit 3 : (extended) Register state
10657                                  <1>
10658 00010B5D F6C320                  <1>       test   bl, 32 ; bit 5
10659 00010B60 0F85B1000000            <1>       jnz    sysvideo_28_7 ; user buffer
10660                                  <1>
10661                                  <1>       ; source or destination is kernel/system buffer
10662                                  <1>
10663 00010B66 803D[80120300]00        <1>       cmp    byte [srvsf], 0 ; srs permission flag
10664 00010B6D 76DB                    <1>       jna    short sysvideo_28_16 ; not permitted
10665                                  <1>
10666 00010B6F F6C301                  <1>       test   bl, 1
10667 00010B72 743A                    <1>       jz     short sysvideo_28_4 ; Save
10668                                  <1>
10669                                  <1>       ; Restore
10670 00010B74 3B0D[82120300]          <1>       cmp    ecx, [VideoStateID]
10671 00010B7A 75CE                    <1>       jne    short sysvideo_28_16 ; not correct ID !
10672                                  <1>
10673 00010B7C 0FB6CA                  <1>       movzx  ecx, dl
10674 00010B7F 80CA80                  <1>       or     dl, 80h
10675 00010B82 3A15[81120300]          <1>       cmp    dl, [srvso]
10676 00010B88 75C0                    <1>       jne    short sysvideo_28_16 ; not correct !
10677                                  <1>
10678 00010B8A 88DA                    <1>       mov    dl, bl
10679                                  <1>
10680                                  <1>       ; ecx = cl = options
10681 00010B8C E8E630FFFF              <1>       call   vbe_srs_gbs
10682                                  <1>       ; ebx = state buffer size (data size)
10683                                  <1>
10684 00010B91 891D[64030300]          <1>       mov    [u.r0], ebx
10685                                  <1>
10686 00010B97 F6C240                  <1>       test   dl, 64 ; verify without transfer
10687 00010B9A 75AE                    <1>       jnz    short sysvideo_28_16 ; yes
10688                                  <1>
10689 00010B9C BE00580900              <1>       mov    esi, VBE3VIDEOSTATE
10690 00010BA1 BF00760900              <1>       mov    edi, VBE3SAVERESTOREBLOCK
10691 00010BA6 87CB                    <1>       xchg   ecx, ebx
10692 00010BA8 F3A4                    <1>       rep    movsb
10693                                  <1>
10694 00010BAA 88D9                    <1>       mov    cl, bl
10695                                  <1>
10696                                  <1>       ; 23/01/2021
10697 00010BAC EB44                    <1>       jmp    short sysvideo_28_10
10698                                  <1>
10699                                  <1> sysvideo_28_4:
10700 00010BAE 53                      <1>       push   ebx
10701                                  <1>       ; 24/01/2021
10702 00010BAF 31DB                    <1>       xor    ebx, ebx ; 0 ; use kernel's buffer
10703 00010BB1 881D[81120300]          <1>       mov    [srvso], bl ; 0 ; invalidate
10704 00010BB7 891D[82120300]          <1>       mov    [VideoStateID], ebx ; 0 ; invalidate
10705 00010BBD 0FB6CA                  <1>       movzx  ecx, dl ; options
10706 00010BC0 B201                    <1>       mov    dl, 1 ; save state
10707 00010BC2 E890000000              <1>       call   sysvideo_28_11 ; 23/01/2021
10708                                  <1>       ; Note: VBE3 BIOS data save option will be
10709                                  <1>       ;      disabled.. ; 24/01/2021
10710 00010BC7 89CA                    <1>       mov    edx, ecx ; state (save) options
10711 00010BC9 5B                      <1>       pop    ebx
10712                                  <1>
10713 00010BCA 6683F84F                <1>       cmp    ax, 4Fh ; successful ?
10714 00010BCE 7536                    <1>       jne    short sysvideo_28_3 ; no !
10715                                  <1>
10716 00010BD0 F6C340                  <1>       test   bl, 64 ; verify without transfer
10717 00010BD3 7536                    <1>       jnz    short sysvideo_28_6 ; yes
10718                                  <1>
10719                                  <1>       ; ecx = cl = options
10720 00010BD5 E89D30FFFF              <1>       call   vbe_srs_gbs
10721                                  <1>       ; ebx = state buffer size (data size)
10722                                  <1>
10723 00010BDA BE00760900              <1>       mov    esi, VBE3SAVERESTOREBLOCK
10724 00010BDF BF00580900              <1>       mov    edi, VBE3VIDEOSTATE
10725 00010BE4 89D9                    <1>       mov    ecx, ebx
10726 00010BE6 F3A4                    <1>       rep    movsb
10727                                  <1>
10728 00010BE8 88D1                    <1>       mov    cl, dl
10729 00010BEA 80C980                  <1>       or     cl, 80h ; SVGA (VESA VBE) flag
10730                                  <1>       ;mov   [srvso], dl
10731                                  <1>
10732 00010BED E908010000              <1>       jmp    sysvideo_28_15
10733                                  <1>
10734                                  <1>       ; 23/01/2021
10735                                  <1> sysvideo_28_10:
10736                                  <1>       ; CL = VESA VBE3 Save/Restore options
10737                                  <1>
10738 00010BF2 B202                    <1>       mov    dl, 2 ; restore state
10739                                  <1>
10740 00010BF4 E85C000000              <1>       call   sysvideo_28_1
10741                                  <1>
10742 00010BF9 6683F84F                <1>       cmp    ax, 4Fh ; successful ?
10743 00010BFD 7407                    <1>       je     short sysvideo_28_3
10744                                  <1>       ;jmp   short sysvideo_28_9
10745                                  <1>
```

```
10746                                  <1> sysvideo_28_9:
10747                                  <1>       ; return zero size (error) to user
10748 00010BFF 29C0                     <1>       sub    eax, eax
10749                                  <1> sysvideo_28_5:
10750 00010C01 A3[64030300]             <1>       mov    [u.r0], eax
10751                                  <1> sysvideo_28_3:
10752 00010C06 E901CDFFFF               <1>       jmp    sysret
10753                                  <1>
10754                                  <1> sysvideo_28_6:
10755                                  <1>       ; use timer ticks as VideoStateID
10756 00010C0B A1[988A0100]             <1>       mov    eax, [TIMER_LH]
10757 00010C10 09C0                     <1>       or     eax, eax
10758 00010C12 75ED                     <1>       jnz    short sysvideo_28_5
10759 00010C14 40                       <1>       inc    eax
10760 00010C15 EBEA                     <1>       jmp    short sysvideo_28_5
10761                                  <1>
10762                                  <1> sysvideo_28_7:
10763                                  <1>       ; save/restore to/from user buffer
10764                                  <1>
10765                                  <1>       ; 23/01/2021
10766 00010C17 89CE                     <1>       mov    esi, ecx ; user's vstate buffer
10767 00010C19 BF00760900               <1>       mov    edi, VBE3SAVERESTOREBLOCK
10768                                  <1>
10769 00010C1E 0FB6CA                   <1>       movzx  ecx, dl ; VESA VBE func 4F04h options
10770                                  <1>
10771                                  <1>       ; source or destination is user buffer
10772 00010C21 F6C301                   <1>       test   bl, 1
10773 00010C24 7444                     <1>       jz     short sysvideo_28_12  ; Save
10774                                  <1>
10775                                  <1>       ; Restore
10776 00010C26 803D[80120300]00         <1>       cmp    byte [srvsf], 0 ; srs permission flag
10777 00010C2D 766A                     <1>       jna    short sysvideo_28_14 ; not permitted
10778                                  <1>
10779 00010C2F 88DA                     <1>       mov    dl, bl ; 'sysvideo' options
10780                                  <1>
10781                                  <1>       ; ecx = cl = options
10782 00010C31 E84130FFFF               <1>       call   vbe_srs_gbs
10783                                  <1>       ; ebx = state buffer size (data size)
10784                                  <1>
10785 00010C36 891D[64030300]           <1>       mov    [u.r0], ebx ; transfer count
10786                                  <1>
10787 00010C3C F6C240                   <1>       test   dl, 64 ; verify without transfer
10788 00010C3F 7558                     <1>       jnz    short sysvideo_28_14 ; yes
10789                                  <1>
10790 00010C41 6681FB0008               <1>       cmp    bx, 2048
10791 00010C46 73B7                     <1>       jnb    short sysvideo_28_9 ; invalid
10792                                  <1>
10793 00010C48 87CB                     <1>       xchg   ecx, ebx
10794                                  <1>       ; esi = user buffer
10795                                  <1>       ; edi = VBE3SAVERESTOREBLOCK
10796                                  <1>
10797 00010C4A E8420F0000               <1>       call   transfer_from_user_buffer
10798 00010C4F 72AE                     <1>       jc     short sysvideo_28_9 ; error
10799                                  <1>
10800 00010C51 89D9                     <1>       mov    ecx, ebx ; Function 4F04h options
10801 00010C53 EB9D                     <1>       jmp    short sysvideo_28_10 ; 23/01/2021
10802                                  <1>
10803                                  <1> sysvideo_28_1:
10804 00010C55 31DB                     <1>       xor    ebx, ebx ; 0 ; use kernel's buffer
10805                                  <1> sysvideo_28_11:
10806                                  <1>       ; 24/01/2021
10807 00010C57 803D[5C090000]03         <1>       cmp    byte [vbe3], 3
10808 00010C5E 7405                     <1>       je     short sysvideo_28_2
10809                                  <1>
10810                                  <1>       ; VESA VBE2 (BOCHS/QEMU/VBOX) video bios
10811 00010C60 E97B2FFFFF               <1>       jmp    _vbe_biosfn_save_restore_state
10812                                  <1> sysvideo_28_2:
10813                                  <1>       ;24/01/2021
10814                                  <1>       ;mov   eax, 4F04h  ; Save/Restore vstate
10815                                  <1>       ; VESA VBE3 video bios
10816 00010C65 E9D40DFFFF               <1>       jmp    _vbe3_pmfn_save_restore_state
10817                                  <1>
10818                                  <1> sysvideo_28_12:
10819                                  <1>       ; Save
10820                                  <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
10821                                  <1>
10822                                  <1>       ;movzx ecx, dl ; options
10823 00010C6A 56                       <1>       push   esi
10824 00010C6B 53                       <1>       push   ebx
10825                                  <1>       ; 23/01/2021
10826 00010C6C B201                     <1>       mov    dl, 1 ; save state
10827 00010C6E E8E2FFFFFF               <1>       call   sysvideo_28_1
10828 00010C73 5A                       <1>       pop    edx ; 'sysvideo' options
10829 00010C74 5F                       <1>       pop    edi ; user's video state buffer
10830                                  <1>
10831 00010C75 6683F84F                 <1>       cmp    ax, 4Fh ; successful ?
10832 00010C79 751E                     <1>       jne    short sysvideo_28_14 ; no !
10833                                  <1>
10834                                  <1>       ; ecx = cl = options
10835 00010C7B E8F72FFFFF               <1>       call   vbe_srs_gbs
10836                                  <1>       ; ebx = state buffer size (data size)
10837                                  <1>
10838 00010C80 89D9                     <1>       mov    ecx, ebx ; transfer count
10839                                  <1>
10840 00010C82 F6C240                   <1>       test   dl, 64 ; verify without transfer
10841 00010C85 750C                     <1>       jnz    short sysvideo_28_13 ; yes
10842                                  <1>
10843                                  <1>       ;mov   edi, esi
10844 00010C87 BE00760900               <1>       mov    esi, VBE3SAVERESTOREBLOCK
10845 00010C8C E8B60E0000               <1>       call   transfer_to_user_buffer
10846 00010C91 7206                     <1>       jc     short sysvideo_28_14
10847                                  <1> sysvideo_28_13:
10848 00010C93 890D[64030300]           <1>       mov    [u.r0], ecx
10849                                  <1> sysvideo_28_14:
10850 00010C99 E96ECCFFFF               <1>       jmp    sysret
```

```
10851                                  <1>
10852                                  <1> sysvideo_29:
10853                                  <1>       ; 18/01/2021
10854                                  <1>       ; BH = 13
10855                                  <1>       ; Save/Restore std VGA video state
10856                                  <1>
10857                                  <1>       ; bl = 0..3
10858                                  <1>       ;      save to or restore from
10859                                  <1>       ;      system buffer, VBE3VIDEOSTATE
10860                                  <1>       ;      ECX = VideoStateID for restoring
10861                                  <1>       ; bl = 4..7
10862                                  <1>       ;      save to or restore from
10863                                  <1>       ;      user buffer pointed by ECX
10864                                  <1>
10865 00010C9E 80FB03                 <1>       cmp    bl, 3
10866 00010CA1 7776                   <1>       ja     short sysvideo_29_6
10867                                  <1>
10868                                  <1>       ; source or destination is kernel/system buffer
10869                                  <1>
10870 00010CA3 803D[80120300]00       <1>       cmp    byte [srvsf], 0 ; srs permission flag
10871 00010CAA 7668                   <1>       jna    short sysvideo_29_5 ; not permitted
10872                                  <1>
10873 00010CAC F6C301                 <1>       test   bl, 1
10874 00010CAF 7437                   <1>       jz     short sysvideo_29_2  ; Save
10875                                  <1>
10876                                  <1>       ; Restore
10877 00010CB1 3B0D[82120300]         <1>       cmp    ecx, [VideoStateID]
10878 00010CB7 755B                   <1>       jne    short sysvideo_29_5 ; not correct !
10879 00010CB9 80FB01                 <1>       cmp    bl, 1
10880 00010CBC 7709                   <1>       ja     short sysvideo_29_0
10881                                  <1>       ; bl = 1
10882 00010CBE BB6E000000             <1>       mov    ebx, 110
10883 00010CC3 B103                   <1>       mov    cl, 3 ; ctrl, vbios data
10884 00010CC5 EB07                   <1>       jmp    short sysvideo_29_1
10885                                  <1> sysvideo_29_0:
10886                                  <1>       ; bl  = 3
10887 00010CC7 BB72030000             <1>       mov    ebx, 882
10888 00010CCC B107                   <1>       mov    cl, 7 ; ctrl, vbios data, dac
10889                                  <1> sysvideo_29_1:
10890 00010CCE 3A0D[81120300]         <1>       cmp    cl, [srvso]
10891 00010CD4 753E                   <1>       jne    short sysvideo_29_5 ; not correct !
10892                                  <1>
10893 00010CD6 BE00580900             <1>       mov    esi, VBE3VIDEOSTATE ; 22/01/2021
10894 00010CDB E83131FFFF             <1>       call   biosfn_restore_video_state
10895 00010CE0 891D[64030300]         <1>       mov    [u.r0], ebx ; video state size (bytes)
10896                                  <1>       ;jmp   sysret
10897 00010CE6 EB2C                   <1>       jmp    short sysvideo_29_5
10898                                  <1> sysvideo_29_2:
10899                                  <1>       ;mov   esi, ecx
10900 00010CE8 BF00580900             <1>       mov    edi, VBE3VIDEOSTATE
10901                                  <1>
10902 00010CED B107                   <1>       mov    cl, 7 ; ctrl, vbios data, dac
10903 00010CEF 08DB                   <1>       or     bl, bl
10904 00010CF1 7502                   <1>       jnz    short sysvideo_29_3 ; bl = 2
10905                                  <1>       ; bl = 0
10906 00010CF3 B103                   <1>       mov    cl, 3 ; ctrl, vbios data
10907                                  <1> sysvideo_29_3:
10908 00010CF5 E8A92FFFFF             <1>       call   biosfn_save_video_state
10909                                  <1> sysvideo_28_15:
10910                                  <1>       ; use timer ticks as VideoStateID
10911 00010CFA A1[988A0100]           <1>       mov    eax, [TIMER_LH]
10912 00010CFF 21C0                   <1>       and    eax, eax
10913 00010D01 7501                   <1>       jnz    short sysvideo_29_4
10914 00010D03 40                     <1>       inc    eax
10915                                  <1> sysvideo_29_4:
10916 00010D04 880D[81120300]         <1>       mov    [srvso], cl
10917 00010D0A A3[82120300]           <1>       mov    [VideoStateID], eax
10918 00010D0F A3[64030300]           <1>       mov    [u.r0], eax
10919                                  <1> sysvideo_29_5:
10920 00010D14 E9F3CBFFFF             <1>       jmp    sysret
10921                                  <1>
10922                                  <1> sysvideo_29_6:
10923 00010D19 80FB07                 <1>       cmp    bl, 7
10924 00010D1C 77F6                   <1>       ja     short sysvideo_29_5 ; invalid sub function
10925                                  <1>
10926 00010D1E 89CE                   <1>       mov    esi, ecx
10927 00010D20 BF00760900             <1>       mov    edi, VBE3SAVERESTOREBLOCK
10928                                  <1>
10929                                  <1>       ; source or destination is user buffer
10930 00010D25 F6C301                 <1>       test   bl, 1
10931 00010D28 7434                   <1>       jz     short sysvideo_29_9  ; Save
10932                                  <1>
10933                                  <1>       ; Restore
10934 00010D2A 803D[80120300]00       <1>       cmp    byte [srvsf], 0 ; srs permission flag
10935 00010D31 76E1                   <1>       jna    short sysvideo_29_5 ; not permitted
10936                                  <1>
10937                                  <1>       ;mov   esi, ecx
10938                                  <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
10939                                  <1>
10940 00010D33 80FB07                 <1>       cmp    bl, 7
10941 00010D36 7409                   <1>       je     short sysvideo_29_7
10942                                  <1>       ; bl = 5
10943 00010D38 B303                   <1>       mov    bl, 3
10944 00010D3A B96E000000             <1>       mov    ecx, 110
10945 00010D3F EB05                   <1>       jmp    short sysvideo_29_8
10946                                  <1> sysvideo_29_7:
10947                                  <1>       ; bl = 7
10948 00010D41 B972030000             <1>       mov    ecx, 882
10949                                  <1> sysvideo_29_8:
10950 00010D46 E8460E0000             <1>       call   transfer_from_user_buffer
10951 00010D4B 72C7                   <1>       jc     short sysvideo_29_5
10952 00010D4D 890D[64030300]         <1>       mov    [u.r0], ecx
10953 00010D53 88D9                   <1>       mov    cl, bl ; mov cl,7 (mov cl,3)
10954 00010D55 89FE                   <1>       mov    esi, edi ; VBE3SAVERESTOREBLOCK
10955                                  <1>       ; cl = 3 or 7
```

```
10956 00010D57 E8B530FFFF          <1>       call   biosfn_restore_video_state
10957 00010D5C EBB6                <1>       jmp    sysvideo_29_5
10958                              <1>       ;jmp   sysret
10959                              <1> sysvideo_29_9:
10960                              <1>       ; Save
10961                              <1>       ;mov   edi, VBE3SAVERESTOREBLOCK
10962                              <1>
10963 00010D5E 80FB06              <1>       cmp    bl, 6
10964 00010D61 7409                <1>       je     short sysvideo_29_10
10965                              <1>       ; bl = 4
10966 00010D63 BB6E000000          <1>       mov    ebx, 110
10967 00010D68 B103                <1>       mov    cl, 3 ; ctrl, vbios data
10968 00010D6A EB07                <1>       jmp    short sysvideo_29_11
10969                              <1> sysvideo_29_10:
10970                              <1>       ; bl = 6
10971 00010D6C BB72030000          <1>       mov    ebx, 882
10972 00010D71 B107                <1>       mov    cl, 7 ; ctrl, vbios data, dac
10973                              <1> sysvideo_29_11:
10974 00010D73 E82B2FFFFF          <1>       call   biosfn_save_video_state
10975                              <1>
10976 00010D78 89D9                <1>       mov    ecx, ebx ; transfer count
10977 00010D7A 89F7                <1>       mov    edi, esi
10978 00010D7C BE00760900          <1>       mov    esi, VBE3SAVERESTOREBLOCK
10979                              <1>
10980                              <1>       ;call  transfer_to_user_buffer
10981                              <1>       ;jc    short sysvideo_29_5
10982                              <1>       ;mov   [u.r0], ecx ; transfer count
10983                              <1>       ;;jmp  sysret
10984                              <1>       ;jmp   short sysvideo_29_5
10985                              <1>
10986 00010D81 EB1A                <1>       jmp    short sysvideo_29_12
10987                              <1>
10988                              <1> sysvideo_30:
10989 00010D83 80FF0F              <1>       cmp    bh, 15
10990 00010D86 7722                <1>       ja     short sysvideo_31 ; invalid function
10991                              <1>
10992                              <1>       ; BH = 15
10993                              <1>       ; Copy VESA EDID to user's buffer
10994                              <1>
10995 00010D88 803D[37430000]4F    <1>       cmp    byte [edid], 4Fh
10996 00010D8F 7519                <1>       jne    short sysvideo_31 ; not ready !
10997                              <1>
10998                              <1>       ;and   ecx, ecx
10999                              <1>       ;jz    short sysvideo_31
11000                              <1>
11001                              <1>       ; ecx = user's buffer address
11002 00010D91 89CF                <1>       mov    edi, ecx
11003 00010D93 BE[9C110300]        <1>       mov    esi, edid_info
11004 00010D98 B980000000          <1>       mov    ecx, 128 ; 128 bytes
11005                              <1> sysvideo_29_12:
11006 00010D9D E8A50D0000          <1>       call   transfer_to_user_buffer
11007 00010DA2 7206                <1>       jc     short sysvideo_31
11008                              <1>
11009 00010DA4 890D[64030300]      <1>       mov    [u.r0], ecx ; EDID size, 128 bytes
11010                              <1> sysvideo_31:
11011 00010DAA E95DCBFFFF          <1>       jmp    sysret
11012                              <1>
11013                              <1> mkdir:
11014                              <1>       ; 04/12/2015 (14 byte directory names)
11015                              <1>       ; 12/10/2015
11016                              <1>       ; 17/06/2015 (Retro UNIX 386 v1 - Beginning)
11017                              <1>       ; 29/04/2013 - 01/08/2013 (Retro UNIX 8086 v1)
11018                              <1>       ;
11019                              <1>       ; 'mkdir' makes a directory entry from the name pointed to
11020                              <1>       ; by u.namep into the current directory.
11021                              <1>       ;
11022                              <1>       ; INPUTS ->
11023                              <1>       ;    u.namep - points to a file name
11024                              <1>       ;                 that is about to be a directory entry.
11025                              <1>       ;    ii - current directory's i-number.
11026                              <1>       ; OUTPUTS ->
11027                              <1>       ;    u.dirbuf+2 - u.dirbuf+10 - contains file name.
11028                              <1>       ;    u.off - points to entry to be filled
11029                              <1>       ;          in the current directory
11030                              <1>       ;    u.base - points to start of u.dirbuf.
11031                              <1>       ;    r1 - contains i-number of current directory
11032                              <1>       ;
11033                              <1>       ; ((AX = R1)) output
11034                              <1>       ;
11035                              <1>       ;    (Retro UNIX Prototype : 11/11/2012, UNIXCOPY.ASM)
11036                              <1>         ;    ((Modified registers: eAX, eDX, eBX, eCX, eSI, eDI, eBP))
11037                              <1>       ;
11038                              <1>
11039                              <1>       ; 17/06/2015 - 32 bit modifications (Retro UNIX 386 v1)
11040 00010DAF 31C0                <1>       xor    eax, eax
11041 00010DB1 BF[9A030300]        <1>       mov     edi, u.dirbuf+2
11042 00010DB6 89FE                <1>       mov    esi, edi
11043 00010DB8 AB                  <1>       stosd
11044 00010DB9 AB                  <1>       stosd
11045                              <1>       ; 04/12/2015 (14 byte directory names)
11046 00010DBA AB                  <1>       stosd
11047 00010DBB 66AB                <1>       stosw
11048                              <1>             ; jsr r0,copyz; u.dirbuf+2; u.dirbuf+10. / clear this
11049 00010DBD 89F7                <1>       mov    edi, esi ; offset to u.dirbuf
11050                              <1>       ; 12/10/2015 ([u.namep] -> ebp)
11051                              <1>       ;mov   ebp, [u.namep]
11052 00010DBF E80D030000          <1>       call   trans_addr_nmbp ; convert virtual address to physical
11053                              <1>             ; esi = physical address (page start + offset)
11054                              <1>             ; ecx = byte count in the page (1 - 4096)
11055                              <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
11056                              <1>             ; mov u.namep,r2 / r2 points to name of directory entry
11057                              <1>             ; mov $u.dirbuf+2,r3 / r3 points to u.dirbuf+2
11058                              <1> mkdir_1: ; 1:
11059 00010DC4 45                  <1>       inc    ebp ; 12/10/2015
11060                              <1>       ;
```

```
11061                                  <1>       ; / put characters in the directory name in u.dirbuf+2 - u.dirbuf+10
11062                                  <1>        ; 01/08/2013
11063 00010DC5 AC                      <1>       lodsb
11064                                  <1>             ; movb (r2)+,r1 / move character in name to r1
11065 00010DC6 20C0                    <1>       and    al, al
11066 00010DC8 7427                    <1>       jz     short mkdir_3
11067                                  <1>             ; beq 1f / if null, done
11068 00010DCA 3C2F                    <1>       cmp    al, '/'
11069                                  <1>             ; cmp r1,$'/ / is it a "/"?
11070 00010DCC 7414                    <1>       je     short mkdir_err
11071                                  <1>       ;je    error
11072                                  <1>             ; beq error9 / yes, error
11073                                  <1>       ; 12/10/2015
11074 00010DCE 6649                    <1>       dec    cx
11075 00010DD0 7505                    <1>       jnz    short mkdir_2
11076                                  <1>       ; 12/10/2015 ([u.namep] -> ebp)
11077 00010DD2 E800030000              <1>       call   trans_addr_nm ; convert virtual address to physical
11078                                  <1>             ; esi = physical address (page start + offset)
11079                                  <1>             ; ecx = byte count in the page
11080                                  <1>       ; edi = offset to u.dirbuf (edi is not modified in trans_addr_nm)
11081                                  <1> mkdir_2:
11082 00010DD7 81FF[A8030300]          <1>       cmp     edi, u.dirbuf+16 ; ; 04/12/2015 (10 -> 16)
11083                                  <1>             ; cmp r3,$u.dirbuf+10. / have we reached the last slot for
11084                                  <1>                                    ; / a char?
11085 00010DDD 74E5                    <1>       je     short mkdir_1
11086                                  <1>             ; beq 1b / yes, go back
11087 00010DDF AA                      <1>       stosb
11088                                  <1>             ; movb r1,(r3)+ / no, put the char in the u.dirbuf
11089 00010DE0 EBE2                    <1>       jmp    short mkdir_1
11090                                  <1>             ; br 1b / get next char
11091                                  <1> mkdir_err:
11092                                  <1>       ; 17/06/2015
11093 00010DE2 C705[C8030300]1300-     <1>       mov    dword [u.error], ERR_NOT_DIR ; 'not a valid directory !'
11093 00010DEA 0000                    <1>
11094 00010DEC E9FBCAFFFF              <1>       jmp    error
11095                                  <1>
11096                                  <1> mkdir_3: ; 1:
11097 00010DF1 A1[78030300]            <1>       mov    eax, [u.dirp]
11098 00010DF6 A3[80030300]            <1>       mov    [u.off], eax
11099                                  <1>             ; mov u.dirp,u.off / pointer to empty current directory
11100                                  <1>                                ; / slot to u.off
11101                                  <1> wdir: ; 29/04/2013
11102 00010DFB C705[84030300]-         <1>         mov     dword [u.base], u.dirbuf
11102 00010E01 [98030300]              <1>
11103                                  <1>             ; mov $u.dirbuf,u.base / u.base points to created file name
11104 00010E05 C705[88030300]1000-     <1>         mov     dword [u.count], 16 ; 04/12/2015 (10 -> 16)
11104 00010E0D 0000                    <1>
11105                                  <1>             ; mov $10.,u.count / u.count = 10
11106 00010E0F 66A1[51040300]          <1>       mov    ax, [ii]
11107                                  <1>             ; mov ii,r1 / r1 has i-number of current directory
11108 00010E15 B201                    <1>       mov    dl, 1 ; owner flag mask ; RETRO UNIX 8086 v1 modification !
11109 00010E17 E8741D0000              <1>       call   access
11110                                  <1>             ; jsr r0,access; 1 / get i-node and set its file up
11111                                  <1>                               ; / for writing
11112                                  <1>       ; AX = i-number of current directory
11113                                  <1>       ; 01/08/2013
11114 00010E1C FE05[C6030300]          <1>       inc     byte [u.kcall] ; the caller is 'mkdir' sign
11115 00010E22 E8820F0000              <1>       call   writei
11116                                  <1>             ; jsr r0,writei / write into directory
11117 00010E27 C3                      <1>       retn
11118                                  <1>             ; rts r0
11119                                  <1>
11120                                  <1> sysexec:
11121                                  <1>       ; 18/11/2017
11122                                  <1>       ; 14/11/2017
11123                                  <1>       ; 13/11/2017
11124                                  <1>       ; 24/10/2016, 04/01/2017
11125                                  <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
11126                                  <1>       ; 23/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
11127                                  <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
11128                                  <1>       ;
11129                                  <1>       ; 'sysexec' initiates execution of a file whose path name if
11130                                  <1>       ; pointed to by 'name' in the sysexec call.
11131                                  <1>       ; 'sysexec' performs the following operations:
11132                                  <1>       ;    1. obtains i-number of file to be executed via 'namei'.
11133                                  <1>       ;    2. obtains i-node of file to be exceuted via 'iget'.
11134                                  <1>       ;    3. sets trap vectors to system routines.
11135                                  <1>       ;    4. loads arguments to be passed to executing file into
11136                                  <1>       ;      highest locations of user's core
11137                                  <1>       ;    5. puts pointers to arguments in locations immediately
11138                                  <1>       ;      following arguments.
11139                                  <1>       ;    6.      saves number of arguments in next location.
11140                                  <1>       ;    7. intializes user's stack area so that all registers
11141                                  <1>       ;      will be zeroed and the PS is cleared and the PC set
11142                                  <1>       ;      to core when 'sysret' restores registers
11143                                  <1>       ;      and does an rti.
11144                                  <1>       ;    8. inializes u.r0 and u.sp
11145                                  <1>       ;    9. zeros user's core down to u.r0
11146                                  <1>       ;   10.      reads executable file from storage device into core
11147                                  <1>       ;      starting at location 'core'.
11148                                  <1>       ;   11.      sets u.break to point to end of user's code with
11149                                  <1>       ;      data area appended.
11150                                  <1>       ;   12.      calls 'sysret' which returns control at location
11151                                  <1>       ;      'core' via 'rti' instruction.
11152                                  <1>       ;
11153                                  <1>       ; Calling sequence:
11154                                  <1>       ;      sysexec; namep; argp
11155                                  <1>       ; Arguments:
11156                                  <1>       ;      namep - points to pathname of file to be executed
11157                                  <1>       ;      argp  - address of table of argument pointers
11158                                  <1>       ;      argp1... argpn - table of argument pointers
11159                                  <1>       ;      argp1:<...0> ... argpn:<...0> - argument strings
11160                                  <1>       ; Inputs: (arguments)
11161                                  <1>       ; Outputs: -
11162                                  <1>       ; ..........................................................
```

```
11163                                  <1>       ;
11164                                  <1>       ; Retro UNIX 386 v1 modification:
11165                                  <1>       ;     User application runs in it's own virtual space
11166                                  <1>       ;     which is izolated from kernel memory (and other
11167                                  <1>       ;     memory pages) via 80386   paging in ring 3
11168                                  <1>       ;     privilige mode. Virtual start address is always 0.
11169                                  <1>       ;     User's core memory starts at linear address 400000h
11170                                  <1>       ;     (the end of the 1st 4MB).
11171                                  <1>       ;
11172                                  <1>       ; Retro UNIX 8086 v1 modification:
11173                                  <1>       ;     user/application segment and system/kernel segment
11174                                  <1>       ;     are different and sysenter/sysret/sysrele routines
11175                                  <1>       ;     are different (user's registers are saved to
11176                                  <1>       ;     and then restored from system's stack.)
11177                                  <1>       ;
11178                                  <1>       ;     NOTE: Retro UNIX 8086 v1 'arg2' routine gets these
11179                                  <1>       ;           arguments which were in these registers;
11180                                  <1>       ;           but, it returns by putting the 1st argument
11181                                  <1>       ;           in 'u.namep' and the 2nd argument
11182                                  <1>       ;           on top of stack. (1st argument is offset of the
11183                                  <1>       ;           file/path name in the user's program segment.)
11184                                  <1>
11185                                  <1>       ;call  arg2
11186                                  <1>       ; * name - 'u.namep' points to address of file/path name
11187                                  <1>       ;          in the user's program segment ('u.segmnt')
11188                                  <1>       ;          with offset in BX register (as sysopen argument 1).
11189                                  <1>       ; * argp - sysexec argument 2 is in CX register
11190                                  <1>       ;          which is on top of stack.
11191                                  <1>       ;
11192                                  <1>             ; jsr r0,arg2 / arg0 in u.namep,arg1 on top of stack
11193                                  <1>
11194                                  <1>       ; 23/06/2015 (32 bit modifications)
11195                                  <1>
11196                                  <1>       ;; 13/11/2017
11197                                  <1>       ;;mov  [u.namep], ebx ; argument 1
11198                                  <1>         ; 18/10/2015
11199 00010E28 890D[4C040300]          <1>       mov     [argv], ecx  ; * ; argument 2
11200                                  <1>
11201                                  <1>       ; 13/11/2017
11202 00010E2E 89DE                    <1>       mov   esi, ebx
11203 00010E30 E88E210000              <1>       call  set_working_path_x
11204 00010E35 7319                    <1>       jnc   short sysexec_0
11205                                  <1>
11206                                  <1>       ;; 'bad command or file name'
11207                                  <1>       ;mov  eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
11208                                  <1>
11209                                  <1>       ; 'file not found !' error
11210 00010E37 B802000000              <1>       mov   eax, ERR_NOT_FOUND ; 02h ; TRDOS 8086
11211                                  <1> sysexec_not_found_err:
11212                                  <1> sysexec_access_error:
11213                                  <1> sysexec_ext_error:
11214 00010E3C A3[64030300]            <1>       mov   [u.r0], eax
11215 00010E41 A3[C8030300]            <1>       mov   [u.error], eax
11216 00010E46 E84D220000              <1>       call  reset_working_path
11217 00010E4B E99CCAFFFF              <1>       jmp   error
11218                                  <1>
11219                                  <1> sysexec_0:
11220                                  <1>       ; 13/11/2017
11221                                  <1>       ;mov  esi, FindFile_Name
11222 00010E50 66B80018                <1>         mov  ax, 1800h ; Only files
11223 00010E54 E83886FFFF              <1>       call  find_first_file
11224 00010E59 72E1                    <1>       jc    short sysexec_not_found_err ; eax = 2
11225                                  <1>
11226                                  <1>       ; check_ file attributes
11227                                  <1>       ; (attribute bits = 00ADVSHR) ; 18h = Directory+Volume
11228                                  <1>       ; BL = Attributes byte
11229                                  <1>
11230 00010E5B F6C306                  <1>         test bl, 6  ; system file or hidden file (S+H)
11231                                  <1>       ;jz   short sysexec_0ext
11232 00010E5E 7417                    <1>       jz    short sysexec_1 ; yes
11233                                  <1>
11234                                  <1>       ; 13/11/2017
11235                                  <1>       ; /// TRDOS386 permission check for multiuser mode ///
11236                                  <1>       ; SYSTEM file or HIDDEN file !!
11237                                  <1>       ; (Only super user has permission to run this file.)
11238                                  <1>
11239                                  <1>       ; ([u.uid]=0 for super user or root in multiuser mode)
11240                                  <1>       ; ([u.uid]=0 for any users in singleuser mode)
11241 00010E60 803D[B0030300]00        <1>       cmp   byte [u.uid], 0 ; Super User ([u.uid]=0) ?
11242                                  <1>       ;jna  short sysexec_0ext
11243 00010E67 760E                    <1>       jna   short sysexec_1 ; yes
11244                                  <1>
11245                                  <1>       ; 'permission denied !' error
11246 00010E69 B80B000000              <1>         mov  eax, ERR_FILE_ACCESS  ; 11 = ERR_PERM_DENIED
11247 00010E6E EBCC                    <1>         jmp  short sysexec_access_error
11248                                  <1>
11249                                  <1> sysexec_not_exf:
11250                                  <1>       ; 'not executable file !' error
11251 00010E70 B816000000              <1>       mov   eax, ERR_NOT_EXECUTABLE
11252 00010E75 EBC5                    <1>       jmp   sysexec_ext_error
11253                                  <1>
11254                                  <1> ;sysexec_0ext:
11255                                  <1> sysexec_1:
11256                                  <1>       ; 18/11/2017
11257 00010E77 BE[C0930100]            <1>       mov   esi, FindFile_Name
11258                                  <1>       ; 13/11/2017
11259                                  <1>       ; check program file name extension
11260                                  <1>       ; ('.PRG' for current TRDOS version)
11261 00010E7C E8B3A0FFFF              <1>       call  check_prg_filename_ext
11262 00010E81 72ED                    <1>       jc    short sysexec_not_exf
11263                                  <1>
11264                                  <1>       ; 18/11/2017
11265 00010E83 3C50                    <1>       cmp   al, 'P'
11266 00010E85 75E9                    <1>       jne   short sysexec_not_exf
11267                                  <1>
```

```
11268                                  <1>       ; '.PRG' extension is OK.
11269                                  <1>       ; Only '.PRG' files are valid program files
11270                                  <1>       ; for current TRDOS 386 version.
11271                                  <1>
11272 00010E87 8B15[EC930100]          <1>       mov    edx, [FindFile_DirEntry+DirEntry_FileSize]
11273 00010E8D 66A1[E4930100]          <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusHI]
11274 00010E93 C1E010                  <1>       shl    eax, 16
11275 00010E96 66A1[EA930100]          <1>       mov    ax, [FindFile_DirEntry+DirEntry_FstClusLO]
11276                                  <1>       ; EAX = First Cluster number
11277                                  <1>       ; EDX = File Size
11278                                  <1>
11279 00010E9C A3[51040300]            <1>       mov    [ii], eax
11280 00010EA1 8915[55040300]          <1>       mov    [i.size], edx
11281                                  <1>
11282                                  <1> ;sysexec_1:
11283                                  <1>       ; 13/11/2017 - TRDOS 386 (TRDOS v2.0)
11284                                  <1>       ; 24/06/2015 - 23/10/2015 (Retro UNIX 386 v1)
11285                                  <1>         ; Moving arguments to the end of [u.upage]
11286                                  <1>       ; (by regarding page borders in user's memory space)
11287                                  <1>       ;
11288                                  <1>       ; 10/10/2015
11289                                  <1>       ; 21/07/2015
11290 00010EA7 89E5                   <1>       mov    ebp, esp ; (**)
11291                                  <1>       ; 18/10/2015
11292 00010EA9 89EF                   <1>       mov    edi, ebp
11293 00010EAB B900010000             <1>       mov    ecx, MAX_ARG_LEN ; 256
11294                                  <1>       ;sub   edi, MAX_ARG_LEN ; 256
11295 00010EB0 29CF                   <1>       sub    edi, ecx
11296 00010EB2 89FC                   <1>       mov    esp, edi ; *!*
11297 00010EB4 31C0                   <1>       xor    eax, eax
11298 00010EB6 A3[8C030300]           <1>       mov    [u.nread], eax ; 0
11299 00010EBB 66A3[4A040300]         <1>       mov    [argc], ax ; 0 ; 13/11/2017
11300 00010EC1 49                     <1>       dec    ecx ; 256 - 1
11301 00010EC2 890D[88030300]         <1>       mov    [u.count], ecx ; MAX_ARG_LEN - 1 ; 255
11302                                  <1>       ;mov   dword [u.count], MAX_ARG_LEN - 1 ; 255
11303                                  <1> sysexec_2:
11304 00010EC8 8B35[4C040300]          <1>       mov    esi, [argv] ; 18/10/2015
11305 00010ECE E866000000              <1>       call   get_argp
11306 00010ED3 B904000000              <1>       mov    ecx, 4 ; mov ecx, 4
11307                                  <1> sysexec_3:
11308 00010ED8 21C0                   <1>       and    eax, eax
11309 00010EDA 0F846F050000           <1>         jz     sysexec_6
11310                                  <1>       ; 18/10/2015
11311 00010EE0 010D[4C040300]         <1>       add    [argv], ecx ; 4
11312 00010EE6 66FF05[4A040300]       <1>       inc    word [argc]
11313                                  <1>       ;
11314 00010EED A3[84030300]           <1>       mov    [u.base], eax
11315                                  <1>       ; 23/10/2015
11316 00010EF2 66C705[C4030300]00-     <1>       mov    word [u.pcount], 0
11316 00010EFA 00                      <1>
11317                                  <1> sysexec_4:
11318 00010EFB E8E70B0000              <1>       call   cpass ; get a character from user's core memory
11319 00010F00 750E                    <1>         jnz  short sysexec_5
11320                                  <1>              ; (max. 255 chars + null)
11321                                  <1>       ; 18/10/2015
11322 00010F02 28C0                   <1>       sub    al, al
11323 00010F04 AA                     <1>       stosb
11324 00010F05 FF05[8C030300]         <1>       inc    dword [u.nread]
11325 00010F0B E93F050000             <1>       jmp    sysexec_6 ; 24/04/2016
11326                                  <1> sysexec_5:
11327 00010F10 AA                     <1>       stosb
11328 00010F11 20C0                   <1>       and    al, al
11329 00010F13 75E6                   <1>       jnz    short sysexec_4
11330 00010F15 B904000000             <1>       mov    ecx, 4
11331 00010F1A 390D[48040300]         <1>       cmp    [ncount], ecx ; 4
11332 00010F20 72A6                   <1>       jb     short sysexec_2
11333 00010F22 8B35[44040300]         <1>       mov    esi, [nbase]
11334 00010F28 010D[44040300]         <1>       add    [nbase], ecx ; 4
11335 00010F2E 66290D[48040300]       <1>       sub    [ncount], cx
11336 00010F35 8B06                   <1>       mov    eax, [esi]
11337 00010F37 EB9F                   <1>       jmp    short sysexec_3
11338                                  <1>
11339                                  <1> get_argp:
11340                                  <1>       ; 14/11/2017 - TRDOS 386 (TRDOS v2.0)
11341                                  <1>       ; 18/10/2015 (nbase, ncount)
11342                                  <1>       ; 21/07/2015
11343                                  <1>       ; 24/06/2015 (Retro UNIX 386 v1)
11344                                  <1>       ; Get (virtual) address of argument from user's core memory
11345                                  <1>       ;
11346                                  <1>       ; INPUT:
11347                                  <1>       ;      esi = virtual address of argument pointer
11348                                  <1>       ; OUTPUT:
11349                                  <1>       ;      eax = virtual address of argument
11350                                  <1>       ;
11351                                  <1>       ; Modified registers: EAX, EBX, ECX, EDX, ESI
11352                                  <1>       ;
11353 00010F39 833D[BC030300]00       <1>       cmp     dword [u.ppgdir], 0 ; /etc/init ?
11354                                  <1>                                 ; (the caller is kernel)
11355 00010F40 7667                   <1>         jna     short get_argpk
11356                                  <1>       ;
11357 00010F42 89F3                   <1>       mov    ebx, esi
11358 00010F44 E8A753FFFF             <1>       call   get_physical_addr ; get physical address
11359 00010F49 0F8289000000           <1>         jc     get_argp_err
11360 00010F4F A3[44040300]           <1>       mov    [nbase], eax ; physical address
11361 00010F54 66890D[48040300]       <1>       mov    [ncount], cx ; remain byte count in page (1-4096)
11362 00010F5B B804000000             <1>       mov    eax, 4 ; 21/07/2015
11363 00010F60 6639C1                 <1>       cmp    cx, ax ; 4
11364 00010F63 735D                   <1>       jnb    short get_argp2
11365 00010F65 89F3                   <1>       mov    ebx, esi
11366 00010F67 01CB                   <1>       add    ebx, ecx
11367 00010F69 E88253FFFF             <1>       call   get_physical_addr ; get physical address
11368 00010F6E 7268                   <1>       jc     short get_argp_err
11369                                  <1>       ;push  esi
11370 00010F70 89C6                   <1>       mov    esi, eax
11371 00010F72 66870D[48040300]       <1>       xchg   cx, [ncount]
```

```
11372 00010F79 8735[44040300]      <1>       xchg   esi, [nbase]
11373 00010F7F B504                <1>       mov    ch, 4
11374 00010F81 28CD                <1>       sub    ch, cl
11375                              <1> get_argp0:
11376 00010F83 AC                  <1>       lodsb
11377 00010F84 6650                <1>       push   ax
11378 00010F86 FEC9                <1>       dec    cl
11379 00010F88 75F9                <1>         jnz     short get_argp0
11380 00010F8A 8B35[44040300]      <1>       mov    esi, [nbase]
11381                              <1>       ; 21/07/2015
11382 00010F90 0FB6C5              <1>       movzx  eax, ch
11383 00010F93 0105[44040300]      <1>       add    [nbase], eax
11384 00010F99 662905[48040300]    <1>       sub    [ncount], ax
11385                              <1> get_argp1:
11386 00010FA0 AC                  <1>       lodsb
11387 00010FA1 FECD                <1>       dec    ch
11388 00010FA3 7447                <1>         jz      short get_argp3
11389 00010FA5 6650                <1>         push ax
11390 00010FA7 EBF7                <1>       jmp     short get_argp1
11391                              <1> get_argpk:
11392                              <1>       ; Argument is in kernel's memory space
11393 00010FA9 66C705[48040300]00- <1>       mov    word [ncount], PAGE_SIZE ; 4096
11393 00010FB1 10                  <1>
11394 00010FB2 8935[44040300]      <1>       mov    [nbase], esi
11395 00010FB8 8305[44040300]04    <1>       add    dword [nbase], 4
11396 00010FBF 8B06                <1>       mov    eax, [esi] ; virtual addr. = physical addr.
11397 00010FC1 C3                  <1>       retn
11398                              <1> get_argp2:
11399                              <1>       ; 21/07/2015
11400                              <1>       ;mov   eax, 4
11401 00010FC2 8B15[44040300]      <1>       mov    edx, [nbase] ; 18/10/2015
11402 00010FC8 0105[44040300]      <1>       add    [nbase], eax
11403 00010FCE 662905[48040300]    <1>       sub    [ncount], ax
11404                              <1>       ;
11405 00010FD5 8B02                <1>       mov    eax, [edx]
11406 00010FD7 C3                  <1>       retn
11407                              <1> get_argp_err:
11408 00010FD8 A3[C8030300]        <1>       mov    [u.error], eax
11409                              <1>       ; 14/11/2017
11410 00010FDD B801000000          <1>       mov    eax, ERR_BAD_CMD_ARG ; 01h ; TRDOS 8086
11411 00010FE2 A3[64030300]        <1>       mov    [u.r0], eax
11412 00010FE7 E900C9FFFF          <1>       jmp    error
11413                              <1> get_argp3:
11414 00010FEC B103                <1>       mov    cl, 3
11415                              <1> get_argp4:
11416 00010FEE C1E008              <1>       shl    eax, 8
11417 00010FF1 665A                <1>       pop    dx
11418 00010FF3 88D0                <1>       mov    al, dl
11419 00010FF5 E2F7                <1>         loop    get_argp4
11420                              <1>       ;pop   esi
11421 00010FF7 C3                  <1>       retn
11422                              <1>
11423                              <1> sysstat:
11424                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
11425                              <1>       ; temporary !
11426 00010FF8 B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
11427 00010FFD A3[C8030300]        <1>         mov     [u.error], eax
11428 00011002 A3[64030300]        <1>         mov     [u.r0], eax
11429 00011007 E9E0C8FFFF          <1>       jmp    error
11430                              <1>
11431                              <1> sysfstat:
11432                              <1>       ; 13/01/2017 - TRDOS 386 (TRDOS v2.0)
11433                              <1>       ; temporary !
11434 0001100C B801000000          <1>       mov    eax, ERR_INV_FNUMBER ; 'invalid function number !'
11435 00011011 A3[C8030300]        <1>         mov     [u.error], eax
11436 00011016 A3[64030300]        <1>         mov     [u.r0], eax
11437 0001101B E9CCC8FFFF          <1>       jmp    error
11438                              <1>
11439                              <1> fclose:
11440                              <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
11441                              <1>       ;
11442                              <1>       ; 18/06/2015 (Retro UNIX 386 v1 - Beginning)
11443                              <1>       ;            (32 bit offset pointer modification)
11444                              <1>       ; 19/04/2013 - 12/01/2014 (Retro UNIX 8086 v1)
11445                              <1>       ;
11446                              <1>       ; Given the file descriptor (index to the u.fp list)
11447                              <1>       ; 'fclose' first gets the i-number of the file via 'getf'.
11448                              <1>       ; If i-node is active (i-number > 0) the entry in
11449                              <1>       ; u.fp list is cleared. If all the processes that opened
11450                              <1>       ; that file close it, then fsp etry is freed and the file
11451                              <1>       ; is closed. If not a return is taken.
11452                              <1>       ; If the file has been deleted while open, 'anyi' is called
11453                              <1>       ; to see anyone else has it open, i.e., see if it is appears
11454                              <1>       ; in another entry in the fsp table. Upon return from 'anyi'
11455                              <1>       ; a check is made to see if the file is special.
11456                              <1>       ;
11457                              <1>       ; INPUTS ->
11458                              <1>       ;    r1 - contains the file descriptor (value=0,1,2...)
11459                              <1>       ;    u.fp - list of entries in the fsp table
11460                              <1>       ;    fsp - table of entries (4 words/entry) of open files.
11461                              <1>       ; OUTPUTS ->
11462                              <1>       ;    r1 - contains the same file descriptor
11463                              <1>       ;    r2 - contains i-number
11464                              <1>       ;
11465                              <1>       ; ((AX = R1))
11466                              <1>       ; ((Modified registers: eDX, eBX, eCX, eSI, eDI, eBP))
11467                              <1>       ;
11468                              <1>       ; Retro UNIX 8086 v1 modification : CF = 1
11469                              <1>       ;             if i-number of the file is 0. (error)
11470                              <1>       ;
11471                              <1>       ; TRDOS 386 (06/10/2016)
11472                              <1>       ;
11473                              <1>       ; INPUT:
11474                              <1>       ;     EAX = File Handle (File Descriptor, File Index)
11475                              <1>       ;
```

```
11476                                  <1>       ; OUTPUT:
11477                                  <1>       ;      CF = 1 -> File not open !
11478                                  <1>       ;      CF = 0 -> OK!
11479                                  <1>       ;           EBX = File Number (System)
11480                                  <1>       ;           [cdev] = Logical DOS Drive Number
11481                                  <1>       ;           EAX = File Handle/Number (user)
11482                                  <1>       ;
11483                                  <1>       ; Modified Registers: EBX
11484                                  <1>
11485 00011020 50                      <1>       push   eax ; File handle
11486                                  <1>
11487 00011021 E846000000              <1>       call   getf
11488 00011026 0F8245240000            <1>       jc     device_close ; eax = device number
11489                                  <1>
11490 0001102C 80BB[3E9A0100]01        <1>       cmp    byte [ebx+OF_MODE], 1 ; open mode ; 0 = empty entry
11491 00011033 722E                    <1>       jb     short fclose_1            ; 1 = read, 2 = write
11492                                  <1>
11493 00011035 83F801                  <1>       cmp    eax, 1 ; is the first cluster number > 0
11494 00011038 7229                    <1>       jb     short fclose_1 ; no, this is empty entry
11495                                  <1>
11496                                  <1> fclose_0:
11497 0001103A FE8B[529A0100]          <1>       dec    byte [ebx+OF_OPENCOUNT] ; decrement the number of processes
11498                                  <1>                                    ; that have opened the file
11499 00011040 7921                    <1>       jns    short fclose_1 ; jump if not negative (jump if bit 7 is 0)
11500                                  <1>                     ; if all processes haven't closed the file, return
11501                                  <1>       ;
11502                                  <1>       ; eax ; First cluster
11503 00011042 31C0                    <1>       xor    eax, eax ; 0
11504 00011044 8883[3E9A0100]          <1>       mov    [ebx+OF_MODE], al ; 0 = empty entry
11505                                  <1>       ;mov   [ebx+OF_STATUS], al ; 0 = empty entry
11506 0001104A 66C1E302                <1>       shl    bx, 2
11507 0001104E 8983[0C9A0100]          <1>       mov    [ebx+OF_FCLUSTER], eax ; 0
11508 00011054 8983[249B0100]          <1>       mov    [ebx+OF_CCLUSTER], eax ; 0
11509                                  <1>       ;mov   [ebx+OF_CCINDEX], eax ; 0
11510 0001105A A3[74030300]            <1>       mov    [u.fofp], eax ; 0
11511 0001105F 66C1EB02                <1>       shr    bx, 2
11512                                  <1> fclose_1: ; 1:
11513 00011063 58                      <1>       pop    eax ; File handle (File Descriptor, File Index)
11514 00011064 C680[6A030300]00        <1>       mov    byte [eax+u.fp], 0 ; clear that entry in the u.fp list
11515 0001106B C3                      <1>       retn
11516                                  <1>
11517                                  <1> getf:
11518                                  <1>       ; 12/10/2016
11519                                  <1>       ; 11/10/2016
11520                                  <1>       ; 08/10/2016
11521                                  <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
11522                                  <1>       ; / get the device number and the i-number of an open file
11523                                  <1>       ; 13/05/2015
11524                                  <1>       ; 11/05/2015 (Retro UNIX 386 v1 - Beginning)
11525                                  <1>       ; 19/04/2013 - 18/11/2013 (Retro UNIX 8086 v1)
11526                                  <1>       ;
11527 0001106C 89C3                    <1>       mov    ebx, eax
11528                                  <1> getf1:
11529 0001106E 83FB0A                  <1>       cmp    ebx, 10
11530 00011071 730A                    <1>         jnb  short getf2
11531 00011073 8A9B[6A030300]          <1>       mov    bl, [ebx+u.fp]
11532 00011079 08DB                    <1>       or     bl, bl
11533 0001107B 7503                    <1>       jnz    short getf3
11534                                  <1> getf2:
11535                                  <1>       ; 'File not open !' error (ax=0)
11536 0001107D 29C0                    <1>       sub    eax, eax
11537 0001107F C3                      <1>       retn
11538                                  <1> getf3:
11539 00011080 F6C380                  <1>       test   bl, 80h
11540 00011083 7530                    <1>       jnz    short getf5 ; device
11541 00011085 FECB                    <1>       dec    bl ; 0 based
11542 00011087 8A83[349A0100]          <1>       mov    al, [ebx+OF_DRIVE]
11543 0001108D A2[46030300]            <1>       mov    [cdev], al
11544 00011092 C0E302                  <1>       shl    bl, 2 ; *4 (dword offset)
11545 00011095 8B83[849A0100]          <1>       mov    eax, [ebx+OF_SIZE]
11546 0001109B A3[55040300]            <1>       mov    [i.size], eax ; file size
11547 000110A0 8D83[5C9A0100]          <1>       lea    eax, [ebx+OF_POINTER] ;12/10/2016
11548 000110A6 A3[74030300]            <1>       mov    [u.fofp], eax
11549 000110AB 8B83[0C9A0100]          <1>       mov    eax, [ebx+OF_FCLUSTER]
11550 000110B1 C0EB02                  <1>       shr    bl, 2 ; /4 (byte offset)
11551                                  <1> getf4:
11552 000110B4 C3                      <1>       retn
11553                                  <1> getf5:
11554                                  <1>       ; get device number
11555 000110B5 80E37F                  <1>       and    bl, 7Fh ; 1 to 7Fh
11556 000110B8 FECB                    <1>       dec    bl ; 0 based (0 to 7Eh)
11557 000110BA 8A83[66980100]          <1>       mov    al, [ebx+DEV_DRIVER]
11558 000110C0 8AAB[D0970100]          <1>       mov    ch, [ebx+DEV_ACCESS]
11559 000110C6 8A8B[84980100]          <1>       mov    cl, [ebx+DEV_OPENMODE]
11560 000110CC 80E5FE                  <1>       and    ch, 0FEh ; reset bit 0 ; dev_close
11561 000110CF F9                      <1>       stc ; cf = 1
11562 000110D0 C3                      <1>       retn
11563                                  <1>
11564                                  <1> trans_addr_nmbp:
11565                                  <1>       ; 18/10/2015
11566                                  <1>       ; 12/10/2015
11567 000110D1 8B2D[7C030300]          <1>       mov    ebp, [u.namep]
11568                                  <1> trans_addr_nm:
11569                                  <1>       ; Convert virtual (pathname) address to physical address
11570                                  <1>       ; (Retro UNIX 386 v1 feature only !)
11571                                  <1>       ; 18/10/2015
11572                                  <1>       ; 12/10/2015 (u.pnbase & u.pncount has been removed from code)
11573                                  <1>       ; 02/07/2015
11574                                  <1>       ; 17/06/2015
11575                                  <1>       ; 16/06/2015
11576                                  <1>       ;
11577                                  <1>       ; INPUTS:
11578                                  <1>       ;      ebp = pathname address (virtual) ; [u.namep]
11579                                  <1>       ;      [u.pgdir] = user's page directory
11580                                  <1>       ; OUTPUT:
```

```
11581                                  <1>       ;       esi = physical address of the pathname
11582                                  <1>       ;     ecx = remain byte count in the page
11583                                  <1>       ;
11584                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX, ESI)
11585                                  <1>       ;
11586 000110D7 833D[BC030300]00        <1>         cmp     dword [u.ppgdir], 0  ; /etc/init ? (sysexec)
11587 000110DE 7618                    <1>       jna   short trans_addr_nmk ; the caller is os kernel;
11588                                  <1>                                  ; it is already physical address
11589 000110E0 50                      <1>       push  eax
11590 000110E1 89EB                    <1>       mov   ebx, ebp ; [u.namep] ; pathname address (virtual)
11591 000110E3 E80852FFFF              <1>             call  get_physical_addr ; get physical address
11592 000110E8 7204                    <1>       jc    short tr_addr_nm_err
11593                                  <1>       ; 18/10/2015
11594                                  <1>       ; eax = physical address
11595                                  <1>       ; cx = remain byte count in page (1-4096)
11596                                  <1>             ; 12/10/2015 (cx = [u.pncount])
11597 000110EA 89C6                    <1>       mov   esi, eax ; 12/10/2015 (esi=[u.pnbase])
11598 000110EC 58                      <1>       pop   eax
11599 000110ED C3                      <1>       retn
11600                                  <1>
11601                                  <1> tr_addr_nm_err:
11602 000110EE A3[C8030300]            <1>       mov   [u.error], eax
11603                                  <1>       ;pop  eax
11604 000110F3 E9F4C7FFFF              <1>       jmp   error
11605                                  <1>
11606                                  <1> trans_addr_nmk:
11607                                  <1>       ; 12/10/2015
11608                                  <1>       ; 02/07/2015
11609 000110F8 8B35[7C030300]          <1>       mov   esi, [u.namep]  ; [u.pnbase]
11610 000110FE 66B90010                <1>       mov   cx, PAGE_SIZE ; 4096 ; [u.pncount]
11611 00011102 C3                      <1>       retn
11612                                  <1>
11613                                  <1> sysbreak:
11614                                  <1>       ; 18/10/2015
11615                                  <1>       ; 07/10/2015
11616                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
11617                                  <1>       ; 20/06/2013 - 24/03/2014 (Retro UNIX 8086 v1)
11618                                  <1>       ;
11619                                  <1>       ; 'sysbreak' sets the programs break points.
11620                                  <1>       ; It checks the current break point (u.break) to see if it is
11621                                  <1>       ; between "core" and the stack (sp). If it is, it is made an
11622                                  <1>       ; even address (if it was odd) and the area between u.break
11623                                  <1>       ; and the stack is cleared. The new breakpoint is then put
11624                                  <1>       ; in u.break and control is passed to 'sysret'.
11625                                  <1>       ;
11626                                  <1>       ; Calling sequence:
11627                                  <1>       ;     sysbreak; addr
11628                                  <1>       ; Arguments: -
11629                                  <1>       ;
11630                                  <1>       ; Inputs: u.break - current breakpoint
11631                                  <1>       ; Outputs: u.break - new breakpoint
11632                                  <1>       ;     area between old u.break and the stack (sp) is cleared.
11633                                  <1>       ; .........................................................
11634                                  <1>       ;
11635                                  <1>       ; Retro UNIX 8086 v1 modification:
11636                                  <1>       ;     The user/application program puts breakpoint address
11637                                  <1>       ;       in BX register as 'sysbreak' system call argument.
11638                                  <1>       ;     (argument transfer method 1)
11639                                  <1>       ;
11640                                  <1>       ;  NOTE: Beginning of core is 0 in Retro UNIX 8086 v1 !
11641                                  <1>       ;     ((!'sysbreak' is not needed in Retro UNIX 8086 v1!))
11642                                  <1>       ;  NOTE:
11643                                  <1>       ;     'sysbreak' clears extended part (beyond of previous
11644                                  <1>       ;     'u.break' address) of user's memory for original unix's
11645                                  <1>       ;     'bss' compatibility with Retro UNIX 8086 v1 (19/11/2013)
11646                                  <1>
11647                                  <1>             ; mov u.break,r1 / move users break point to r1
11648                                  <1>             ; cmp r1,$core / is it the same or lower than core?
11649                                  <1>             ; blos 1f / yes, 1f
11650                                  <1>       ; 23/06/2015
11651 00011103 8B2D[90030300]          <1>       mov   ebp, [u.break] ; virtual address (offset)
11652                                  <1>       ;and  ebp, ebp
11653                                  <1>       ;jz   short sysbreak_3
11654                                  <1>       ; Retro UNIX 386 v1 NOTE: u.break points to virtual address !!!
11655                                  <1>       ; (Even break point address is not needed for Retro UNIX 386 v1)
11656 00011109 8B15[5C030300]          <1>       mov   edx, [u.sp] ; kernel stack at the beginning of sys call
11657 0001110F 83C20C                  <1>       add   edx, 12 ; EIP -4-> CS -4-> EFLAGS -4-> ESP (user)
11658                                  <1>       ; 07/10/2015
11659 00011112 891D[90030300]          <1>       mov   [u.break], ebx ; virtual address !!!
11660                                  <1>       ;
11661 00011118 3B1A                    <1>       cmp   ebx, [edx] ; compare new break point with
11662                                  <1>                        ; with top of user's stack (virtual!)
11663 0001111A 7323                    <1>       jnb   short sysbreak_3
11664                                  <1>             ; cmp r1,sp / is it the same or higher
11665                                  <1>                      ; / than the stack?
11666                                  <1>             ; bhis 1f / yes, 1f
11667 0001111C 89DE                    <1>       mov   esi, ebx
11668 0001111E 29EE                    <1>       sub   esi, ebp ; new break point - old break point
11669 00011120 761D                    <1>       jna   short sysbreak_3
11670                                  <1>       ;push ebx
11671                                  <1> sysbreak_1:
11672 00011122 89EB                    <1>       mov   ebx, ebp
11673 00011124 E8C751FFFF              <1>       call  get_physical_addr ; get physical address
11674 00011129 72C3                    <1>       jc    tr_addr_nm_err
11675                                  <1>       ; 18/10/2015
11676 0001112B 89C7                    <1>       mov   edi, eax
11677 0001112D 29C0                    <1>       sub   eax, eax ; 0
11678                                  <1>             ; ECX = remain byte count in page (1-4096)
11679 0001112F 39CE                    <1>       cmp   esi, ecx
11680 00011131 7302                    <1>       jnb   short sysbreak_2
11681 00011133 89F1                    <1>       mov   ecx, esi
11682                                  <1> sysbreak_2:
11683 00011135 29CE                    <1>       sub   esi, ecx
11684 00011137 01CD                    <1>       add   ebp, ecx
11685 00011139 F3AA                    <1>       rep   stosb
```

```
11686 0001113B 09F6               <1>       or    esi, esi
11687 0001113D 75E3               <1>       jnz   short sysbreak_1
11688                             <1>       ;
11689                             <1>             ; bit $1,r1 / is it an odd address
11690                             <1>             ; beq 2f / no, its even
11691                             <1>             ; clrb (r1)+ / yes, make it even
11692                             <1>       ; 2: / clear area between the break point and the stack
11693                             <1>             ; cmp r1,sp / is it higher or same than the stack
11694                             <1>             ; bhis 1f / yes, quit
11695                             <1>             ; clr (r1)+ / clear word
11696                             <1>             ; br 2b / go back
11697                             <1>       ;pop  ebx
11698                             <1> sysbreak_3: ; 1:
11699                             <1>       ;mov  [u.break], ebx ; virtual address !!!
11700                             <1>             ; jsr r0,arg; u.break / put the "address"
11701                             <1>                   ; / in u.break (set new break point)
11702                             <1>             ; br sysret4 / br sysret
11703 0001113F E9C8C7FFFF          <1>       jmp   sysret
11704                             <1>
11705                             <1> sysseek: ; / moves read write pointer in an fsp entry
11706                             <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
11707                             <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11708                             <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
11709                             <1>       ;
11710                             <1>       ; 'sysseek' changes the r/w pointer of (3rd word of in an
11711                             <1>       ; fsp entry) of an open file whose file descriptor is in u.r0.
11712                             <1>       ; The file descriptor refers to a file open for reading or
11713                             <1>       ; writing. The read (or write) pointer is set as follows:
11714                             <1>       ;     * if 'ptrname' is 0, the pointer is set to offset.
11715                             <1>       ;     * if 'ptrname' is 1, the pointer is set to its
11716                             <1>       ;       current location plus offset.
11717                             <1>       ;     * if 'ptrname' is 2, the pointer is set to the
11718                             <1>       ;       size of file plus offset.
11719                             <1>       ; The error bit (e-bit) is set for an undefined descriptor.
11720                             <1>       ;
11721                             <1>       ; Calling sequence:
11722                             <1>       ;     sysseek; offset; ptrname
11723                             <1>       ; Arguments:
11724                             <1>       ;     offset - number of bytes desired to move
11725                             <1>       ;              the r/w pointer
11726                             <1>       ;     ptrname - a switch indicated above
11727                             <1>       ;
11728                             <1>       ; Inputs: r0 - file descriptor
11729                             <1>       ; Outputs: -
11730                             <1>       ; ..........................................................
11731                             <1>       ;
11732                             <1>       ; Retro UNIX 8086 v1 modification:
11733                             <1>       ;       'sysseek' system call has three arguments; so,
11734                             <1>       ;     * 1st argument, file descriptor is in BX (BL) register
11735                             <1>       ;     * 2nd argument, offset is in CX register
11736                             <1>       ;     * 3rd argument, ptrname/switch is in DX (DL) register
11737                             <1>
11738 00011144 E821000000        <1>       call  seektell
11739                             <1>       ; EAX = Current R/W pointer of the file
11740                             <1>       ; EBX = [u.fofp]
11741                             <1>       ; [u.base] = offset (ECX input)
11742                             <1>
11743 00011149 0305[84030300]     <1>       add   eax, [u.base]
11744 0001114F 8903               <1>       mov   [ebx], eax
11745 00011151 E9B6C7FFFF         <1>       jmp   sysret
11746                             <1>
11747                             <1> systell: ; / get the r/w pointer
11748                             <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0) - temporary !-
11749                             <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11750                             <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
11751                             <1>       ;
11752                             <1>       ; Retro UNIX 8086 v1 modification:
11753                             <1>       ; ! 'systell' does not work in original UNIX v1,
11754                             <1>       ;         it returns with error !
11755                             <1>       ; Inputs: r0 - file descriptor
11756                             <1>       ; Outputs: r0 - file r/w pointer
11757                             <1>
11758                             <1>       ;xor  ecx, ecx ; 0
11759 00011156 BA01000000        <1>       mov   edx, 1 ; 05/08/2013
11760                             <1>       ;call seektell
11761 0001115B E810000000        <1>       call  seektell0 ; 05/08/2013
11762                             <1>       ;; 06/11/2016
11763                             <1>       ;; mov eax, [ebx]
11764 00011160 A3[64030300]       <1>       mov   [u.r0], eax
11765 00011165 E9A2C7FFFF         <1>       jmp   sysret
11766                             <1>
11767                             <1> ; Original unix v1 'systell' system call:
11768                             <1>             ; jsr r0,seektell
11769                             <1>             ; br error4
11770                             <1>
11771                             <1> seektell:
11772                             <1>       ; 06/11/2016 - TRDOS 386 (TRDOS v2.0)
11773                             <1>       ; 03/01/2016
11774                             <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11775                             <1>       ; 07/07/2013 - 05/08/2013 (Retro UNIX 8086 v1)
11776                             <1>       ;
11777                             <1>       ; 'seektell' puts the arguments from sysseek and systell
11778                             <1>       ; call in u.base and u.count. It then gets the i-number of
11779                             <1>       ; the file from the file descriptor in u.r0 and by calling
11780                             <1>       ; getf. The i-node is brought into core and then u.count
11781                             <1>       ; is checked to see it is a 0, 1, or 2.
11782                             <1>       ; If it is 0 - u.count stays the same
11783                             <1>       ;          1 - u.count = offset (u.fofp)
11784                             <1>       ;        2 - u.count = i.size (size of file)
11785                             <1>       ;
11786                             <1>       ; !! Retro UNIX 8086 v1 modification:
11787                             <1>       ;     Argument 1, file descriptor is in BX;
11788                             <1>       ;     Argument 2, offset is in CX;
11789                             <1>       ;     Argument 3, ptrname/switch is in DX register.
11790                             <1>       ;
```

```
11791                                  <1>       ; ((Return -> eax = base for offset (position= base+offset))
11792                                  <1>       ;
11793 0001116A 890D[84030300]          <1>       mov    [u.base], ecx ; offset
11794                                  <1> seektell0:
11795 00011170 8915[88030300]          <1>       mov    [u.count], edx
11796                                  <1>       ; EBX = file descriptor (file number)
11797 00011176 E8F3FEFFFF              <1>       call   getf1
11798                                  <1>       ; EAX = First cluster of the file
11799                                  <1>       ; EBX = File number (Open file number)
11800                                  <1>       ; [u.fofp] = Pointer to File pointer
11801                                  <1>       ; [i.size] = File size
11802                                  <1>
11803 0001117B 09C0                    <1>       or     eax, eax
11804 0001117D 7514                    <1>       jnz    short seektell1
11805                                  <1>
11806 0001117F B80A000000              <1>       mov    eax, ERR_FILE_NOT_OPEN
11807 00011184 A3[64030300]            <1>       mov    [u.r0], eax
11808 00011189 A3[C8030300]            <1>       mov    dword [u.error], eax ; 'file not open !'
11809 0001118E E959C7FFFF              <1>       jmp    error
11810                                  <1>
11811                                  <1> seektell1:
11812 00011193 8B1D[74030300]          <1>         mov     ebx, [u.fofp]
11813 00011199 803D[88030300]01        <1>       cmp    byte [u.count], 1
11814 000111A0 7705                    <1>       ja     short seektell2
11815 000111A2 7409                    <1>       je     short seektell3
11816 000111A4 31C0                    <1>       xor    eax, eax
11817 000111A6 C3                      <1>       retn
11818                                  <1>
11819                                  <1> seektell2:
11820 000111A7 A1[55040300]            <1>         mov        eax, [i.size]
11821 000111AC C3                      <1>       retn
11822                                  <1>
11823                                  <1> seektell3:
11824 000111AD 8B03                    <1>       mov    eax, [ebx]
11825 000111AF C3                      <1>       retn
11826                                  <1>
11827                                  <1> sysintr: ; / set interrupt handling
11828                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11829                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
11830                                  <1>       ;
11831                                  <1>       ; 'sysintr' sets the interrupt handling value. It puts
11832                                  <1>       ; argument of its call in u.intr then branches into 'sysquit'
11833                                  <1>       ; routine. u.tty is checked if to see if a control tty exists.
11834                                  <1>       ; If one does the interrupt character in the tty buffer is
11835                                  <1>       ; cleared and 'sysret'is called. If one does not exits
11836                                  <1>       ; 'sysret' is just called.
11837                                  <1>       ;
11838                                  <1>       ; Calling sequence:
11839                                  <1>       ;     sysintr; arg
11840                                  <1>       ; Argument:
11841                                  <1>       ;     arg - if 0, interrupts (ASCII DELETE) are ignored.
11842                                  <1>       ;         - if 1, intterupts cause their normal result
11843                                  <1>       ;           i.e force an exit.
11844                                  <1>       ;         - if arg is a location within the program,
11845                                  <1>       ;           control is passed to that location when
11846                                  <1>       ;           an interrupt occurs.
11847                                  <1>       ; Inputs: -
11848                                  <1>       ; Outputs: -
11849                                  <1>       ; ..........................................................
11850                                  <1>       ;
11851                                  <1>       ; Retro UNIX 8086 v1 modification:
11852                                  <1>       ;       'sysintr' system call sets u.intr to value of BX
11853                                  <1>       ;     then branches into sysquit.
11854                                  <1>       ;
11855 000111B0 66891D[AA030300]        <1>       mov    [u.intr], bx
11856                                  <1>             ; jsr r0,arg; u.intr / put the argument in u.intr
11857                                  <1>             ; br 1f / go into quit routine
11858 000111B7 E950C7FFFF              <1>       jmp    sysret
11859                                  <1>
11860                                  <1> sysquit:
11861                                  <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11862                                  <1>       ; 07/07/2013 (Retro UNIX 8086 v1)
11863                                  <1>       ;
11864                                  <1>       ; 'sysquit' turns off the quit signal. it puts the argument of
11865                                  <1>       ; the call in u.quit. u.tty is checked if to see if a control
11866                                  <1>       ; tty exists. If one does the interrupt character in the tty
11867                                  <1>       ; buffer is cleared and 'sysret'is called. If one does not exits
11868                                  <1>       ; 'sysret' is just called.
11869                                  <1>       ;
11870                                  <1>       ; Calling sequence:
11871                                  <1>       ;     sysquit; arg
11872                                  <1>       ; Argument:
11873                                  <1>       ;     arg - if 0, this call diables quit signals from the
11874                                  <1>       ;           typewriter (ASCII FS)
11875                                  <1>       ;         - if 1, quits are re-enabled and cause execution to
11876                                  <1>       ;           cease and a core image to be produced.
11877                                  <1>       ;            i.e force an exit.
11878                                  <1>       ;         - if arg is an addres in the program,
11879                                  <1>       ;           a quit causes control to sent to that
11880                                  <1>       ;           location.
11881                                  <1>       ; Inputs: -
11882                                  <1>       ; Outputs: -
11883                                  <1>       ; ..........................................................
11884                                  <1>       ;
11885                                  <1>       ; Retro UNIX 8086 v1 modification:
11886                                  <1>       ;       'sysquit' system call sets u.quit to value of BX
11887                                  <1>       ;     then branches into 'sysret'.
11888                                  <1>       ;
11889 000111BC 66891D[AC030300]        <1>       mov    [u.quit], bx
11890 000111C3 E944C7FFFF              <1>       jmp    sysret
11891                                  <1>             ; jsr r0,arg; u.quit / put argument in u.quit
11892                                  <1>       ;1:
11893                                  <1>             ; mov u.ttyp,r1 / move pointer to control tty buffer
11894                                  <1>                         ; / to r1
11895                                  <1>             ; beq sysret4 / return to user
```

```
11896                                   <1>             ; clrb 6(r1) / clear the interrupt character
11897                                   <1>                     ; / in the tty buffer
11898                                   <1>             ; br sysret4 / return to user
11899                                   <1>
11900                                   <1> anyi:
11901                                   <1>       ; 06/10/2016 (TRDOS 386 = TRDOS v2.0)
11902                                   <1>       ; Major Modification!
11903                                   <1>       ; TRDOS 386 does not permit to delete a file while it is open
11904                                   <1>       ; The role of 'anyi' procedure has beeen changed to ensure that.
11905                                   <1>       ;
11906                                   <1>       ; 22/06/2015 (Retro UNIX 386 v1 - Beginning)
11907                                   <1>       ; 25/04/2013 (Retro UNIX 8086 v1)
11908                                   <1>       ;
11909                                   <1>       ; 'anyi' is called if a file deleted while open.
11910                                   <1>       ; "anyi" checks to see if someone else has opened this file.
11911                                   <1>       ;
11912                                   <1>       ; INPUTS ->
11913                                   <1>       ;    r1 - contains an i-number
11914                                   <1>       ;    fsp - start of table containing open files
11915                                   <1>       ;
11916                                   <1>       ; OUTPUTS ->
11917                                   <1>       ;    "deleted" flag set in fsp entry of another occurrence of
11918                                   <1>       ;        this file and r2 points 1st word of this fsp entry.
11919                                   <1>       ;    if file not found - bit in i-node map is cleared
11920                                   <1>       ;                    (i-node is freed)
11921                                   <1>       ;              all blocks related to i-node are freed
11922                                   <1>       ;             all flags in i-node are cleared
11923                                   <1>       ; ((AX = R1)) input
11924                                   <1>       ;
11925                                   <1>       ;    (Retro UNIX Prototype : 02/12/2012, UNIXCOPY.ASM)
11926                                   <1>         ;    ((Modified registers: eDX, eCX, eBX, eSI, eDI, eBP))
11927                                   <1>       ;
11928                                   <1>       ; / r1 contains an i-number
11929                                   <1>
11930                                   <1>       ; TRDOS 386 (06/10/2016)
11931                                   <1>       ;
11932                                   <1>       ; INPUT:
11933                                   <1>       ;     EAX = First Cluster
11934                                   <1>       ;      DL = Logical DOS Drive Number
11935                                   <1>       ;
11936                                   <1>       ; OUTPUT:
11937                                   <1>       ;     CF = 1 -> EBX = File Handle/Number/Index
11938                                   <1>       ;     CF = 0 -> EBX = 0
11939                                   <1>       ;
11940                                   <1>       ; Modified Registers: EBX
11941                                   <1>
11942 000111C8 31DB                    <1>       xor   ebx, ebx
11943                                   <1> anyi_0:
11944 000111CA 80BB[3E9A0100]00        <1>       cmp   byte [ebx+OF_MODE], 0 ; 0 = empty entry
11945 000111D1 770A                    <1>       ja    short anyi_2 ; 1 (r), 2 (w) or 3 (r&w)
11946                                   <1> anyi_1:
11947 000111D3 FEC3                    <1>       inc   bl
11948 000111D5 80FB0A                  <1>       cmp   bl, OPENFILES ; max. count of open files
11949 000111D8 72F0                    <1>       jb    short anyi_0
11950 000111DA 31C0                    <1>       xor   eax, eax
11951 000111DC C3                      <1>       retn
11952                                   <1> anyi_2:
11953 000111DD 3A93[349A0100]          <1>       cmp   dl, [ebx+OF_DRIVE]
11954 000111E3 75EE                    <1>       jne   short anyi_1
11955 000111E5 66C1E302                <1>       shl   bx, 2 ; *4 (dword offset)
11956 000111E9 3B83[0C9A0100]          <1>       cmp   eax, [ebx+OF_FCLUSTER]
11957 000111EF 7406                    <1>       je    short anyi_3
11958 000111F1 66C1EB02                <1>       shr   bx, 2 ; /4 (byte offset)
11959 000111F5 EBDC                    <1>       jmp   short anyi_1
11960                                   <1> anyi_3:
11961 000111F7 66C1EB02                <1>       shr   bx, 2 ; /4 (bytes offset) (index)
11962 000111FB F9                      <1>       stc
11963 000111FC C3                      <1>       retn
11964                                   <1>
11965                                   <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS9.INC
11966                                   <1> ; Last Modification: 09/12/2015
11967                                   <1>
11968                                   <1> syssleep:
11969                                   <1>       ; 29/06/2015 - (Retro UNIX 386 v1)
11970                                   <1>       ; 11/06/2014 - (Retro UNIX 8086 v1)
11971                                   <1>       ;
11972                                   <1>       ; Retro UNIX 8086 v1 feature only
11973                                   <1>       ; (INPUT -> none)
11974                                   <1>       ;
11975 000111FD 0FB61D[B3030300]        <1>       movzx  ebx, byte [u.uno] ; process number
11976 00011204 8AA3[7F000300]          <1>       mov    ah, [ebx+p.ttyc-1] ; current/console tty
11977 0001120A E881190000              <1>       call   sleep
11978 0001120F E9F8C6FFFF              <1>       jmp    sysret
11979                                   <1>
11980                                   <1> _vp_clr:
11981                                   <1>       ; Reset/Clear Video Page
11982                                   <1>       ;
11983                                   <1>       ; 30/06/2015 - (Retro UNIX 386 v1)
11984                                   <1>       ; 21/05/2013 - 30/10/2013(Retro UNIX 8086 v1) (U0.ASM)
11985                                   <1>       ;
11986                                   <1>       ; Retro UNIX 8086 v1 feature only !
11987                                   <1>       ;
11988                                   <1>       ; INPUTS ->
11989                                   <1>       ;   BH = video page number
11990                                   <1>       ;
11991                                   <1>       ; OUTPUT ->
11992                                   <1>       ;   none
11993                                   <1>       ; ((Modified registers: eAX, BH, eCX, eDX, eSI, eDI))
11994                                   <1>       ;
11995                                   <1>       ; 04/12/2013
11996 00011214 28C0                    <1>       sub    al, al
11997                                   <1>       ; al = 0 (clear video page)
11998                                   <1>       ; bh = video page ; 13/05/2016
11999 00011216 B407                    <1>       mov    ah, 07h
12000                                   <1>       ; ah = 7 (attribute/color)
```

```
12001 00011218 6631C9              <1>       xor    cx, cx ; 0, left upper column (cl) & row (cl)
12002 0001121B 66BA4F18            <1>       mov    dx, 184Fh ; right lower column & row (dl=24, dh=79)
12003 0001121F E8210EFFFF          <1>       call   _scroll_up
12004                              <1>       ; bh = video page
12005 00011224 6631D2              <1>       xor    dx, dx ; 0 (cursor position)
12006 00011227 E97311FFFF          <1>       jmp    _set_cpos
12007                              <1>
12008                              <1> sysmsg:
12009                              <1>       ; 07/12/2020
12010                              <1>       ; 05/12/2020
12011                              <1>       ; 13/05/2016
12012                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
12013                              <1>       ; 01/07/2015 - 11/11/2015 (Retro UNIX 386 v1)
12014                              <1>       ; Print user-application message on user's console tty
12015                              <1>       ;
12016                              <1>       ; Input -> EBX = Message address
12017                              <1>       ;         ECX = Message length (max. 255)
12018                              <1>       ;         DL = Color (IBM PC Rombios color attributes)
12019                              <1>       ;
12020 0001122C 81F9FF000000        <1>       cmp    ecx, MAX_MSG_LEN ; 255
12021 00011232 0F87D4C6FFFF        <1>       ja     sysret ; nothing to do with big message size
12022 00011238 08C9                <1>       or     cl, cl
12023 0001123A 0F84CCC6FFFF        <1>       jz     sysret
12024 00011240 20D2                <1>       and    dl, dl
12025 00011242 7502                <1>       jnz    short sysmsg0
12026 00011244 B207                <1>       mov    dl, 07h ; default color
12027                              <1>              ; (black background, light gray character)
12028                              <1> sysmsg0:
12029 00011246 891D[84030300]      <1>       mov    [u.base], ebx
12030 0001124C 8815[478A0100]      <1>       mov    [ccolor], dl ; color attributes
12031 00011252 89E5                <1>       mov    ebp, esp
12032 00011254 31DB                <1>       xor    ebx, ebx ; 0
12033 00011256 891D[8C030300]      <1>       mov    [u.nread], ebx ; 0
12034                              <1>       ;
12035 0001125C 381D[C6030300]      <1>       cmp    [u.kcall], bl ; 0
12036 00011262 776F                <1>       ja     short sysmsgk ; Temporary (01/07/2015)
12037                              <1>       ;
12038 00011264 890D[88030300]      <1>       mov    [u.count], ecx
12039                              <1>       ;inc   ecx ; + 00h ; ASCIIZ
12040                              <1>       ;
12041                              <1>       ; 07/12/2020
12042                              <1>       ;add   ecx, 3
12043 0001126A 6683C103            <1>       add    cx, 3
12044 0001126E 80E1FC              <1>       and    cl, ~3  ; not 3
12045                              <1>       ;
12046 00011271 29CC                <1>       sub    esp, ecx
12047 00011273 89E7                <1>       mov    edi, esp
12048 00011275 89E6                <1>       mov    esi, esp
12049 00011277 66891D[C4030300]    <1>       mov    [u.pcount], bx ; reset page (phy. addr.) counter
12050                              <1>       ; 11/11/2015
12051 0001127E 8A25[94030300]      <1>       mov    ah, [u.ttyp] ; recent open tty
12052                              <1>       ; 0 = none
12053 00011284 FECC                <1>       dec    ah
12054 00011286 790C                <1>       jns    short sysmsg1
12055 00011288 8A1D[B3030300]      <1>       mov    bl, [u.uno] ; process number
12056 0001128E 8AA3[7F000300]      <1>       mov    ah, [ebx+p.ttyc-1] ; user's (process's) console tty
12057                              <1> sysmsg1:
12058 00011294 8825[96030300]      <1>       mov    [u.ttyn], ah
12059                              <1> sysmsg2:
12060 0001129A E848080000          <1>       call   cpass
12061 0001129F 7416                <1>       jz     short sysmsg5
12062 000112A1 AA                  <1>       stosb
12063 000112A2 20C0                <1>       and    al, al
12064 000112A4 75F4                <1>       jnz    short sysmsg2
12065                              <1> sysmsg3:
12066 000112A6 80FC07              <1>       cmp    ah, 7 ; tty number
12067 000112A9 7711                <1>       ja     short sysmsg6 ; serial port
12068 000112AB E83E000000          <1>       call   print_cmsg ; 05/12/2020
12069                              <1> sysmsg4:
12070 000112B0 89EC                <1>       mov    esp, ebp
12071 000112B2 E955C6FFFF          <1>       jmp    sysret
12072                              <1> sysmsg5:
12073 000112B7 C60700              <1>       mov    byte [edi], 0
12074 000112BA EBEA                <1>       jmp    short sysmsg3
12075                              <1> sysmsg6:
12076 000112BC 8A06                <1>       mov    al, [esi]
12077 000112BE E8CD180000          <1>       call   sndc
12078 000112C3 72EB                <1>       jc     short sysmsg4
12079 000112C5 803E00              <1>       cmp    byte [esi], 0  ; 0 is stop character
12080 000112C8 76E6                <1>       jna    short sysmsg4
12081 000112CA 46                  <1>       inc    esi
12082 000112CB 8A25[96030300]      <1>       mov    ah, [u.ttyn]
12083 000112D1 EBE9                <1>       jmp    short sysmsg6
12084                              <1>
12085                              <1> sysmsgk: ; Temporary (01/07/2015)
12086                              <1>       ; The message has been sent by Kernel (ASCIIZ string)
12087                              <1>       ; (ECX -character count- will not be considered)
12088 000112D3 8B35[84030300]      <1>       mov    esi, [u.base]
12089 000112D9 8A25[468A0100]      <1>       mov    ah, [ptty] ; present/current screen (video page)
12090 000112DF 8825[96030300]      <1>       mov    [u.ttyn], ah
12091 000112E5 C605[C6030300]00    <1>       mov    byte [u.kcall], 0
12092 000112EC EBB8                <1>       jmp    short sysmsg3
12093                              <1>
12094                              <1> print_cmsg:
12095                              <1>       ; 08/12/2020
12096                              <1>       ; 07/12/2020
12097                              <1>       ; 05/12/2020
12098                              <1>       ; 18/11/2017
12099                              <1>       ; 13/05/2016 - TRDOS 386 (TRDOS v2.0)
12100                              <1>       ; 01/07/2015 (Retro UNIX 386 v1)
12101                              <1>       ;
12102                              <1>       ; print message (on user's console tty)
12103                              <1>       ;     with requested color
12104                              <1>       ;
12105                              <1>       ; INPUTS:
```

```
12106                                  <1>       ;      esi = message address
12107                                  <1>       ;      [u.ttyn] = tty number (0 to 7)
12108                                  <1>       ;      [ccolor] = color attributes (IBM PC BIOS colors)
12109                                  <1>       ;
12110                                  <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
12111                                  <1>       ; (ebp must be preserved)
12112                                  <1>
12113                                  <1>       ;mov   bh, ah
12114 000112EE 8A3D[96030300]         <1>       mov    bh, [u.ttyn]
12115 000112F4 8A1D[478A0100]         <1>       mov    bl, [ccolor] ; * ; 05/12/2020
12116                                  <1>
12117                                  <1>       ; 05/12/2020
12118 000112FA 803D[1C120300]00       <1>       cmp    byte [pmi32], 0 ; is vbios's 32 bit pmi enabled ?
12119 00011301 772E                   <1>       ja     short pcmsg5 ; yes
12120                                  <1> pcmsg1:
12121                                  <1>       ; 08/12/2020
12122 00011303 8A1D[478A0100]         <1>       mov    bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
12123                                  <1>
12124 00011309 AC                     <1>       lodsb
12125 0001130A 20C0                   <1>       and    al, al  ; 0
12126 0001130C 743A                   <1>       jz     short pcmsg2
12127                                  <1> pcmsg7:
12128 0001130E 56                     <1>       push   esi
12129                                  <1>       ;mov   bl, [ccolor] ; * (video.s 'u11'&'beep' change BL)
12130                                  <1>       ; 05/12/2020
12131                                  <1>       ;;mov  bh, [u.ttyn]
12132                                  <1>       ;call  _write_tty
12133                                  <1>       ;pop   esi
12134                                  <1>       ;jmp   short pcmsg1
12135                                  <1> ;pcmsg2:
12136                                  <1>       ;retn
12137                                  <1>
12138                                  <1>       ; 07/12/2020
12139 0001130F 803D[DA6F0000]03       <1>       cmp     byte [CRT_MODE], 3
12140 00011316 7708                   <1>       ja     short pcmsg4
12141                                  <1> pcmsg3:
12142 00011318 E8FB0FFFFF             <1>       call   _write_tty_m3
12143 0001131D 5E                     <1>       pop    esi
12144 0001131E EBE3                   <1>       jmp    short pcmsg1
12145                                  <1> pcmsg4:
12146 00011320 803D[DA6F0000]07       <1>         cmp     byte [CRT_MODE], 7
12147 00011327 76EF                   <1>       jna    short pcmsg3
12148 00011329 E80E1DFFFF             <1>       call   vga_write_teletype
12149 0001132E 5E                     <1>       pop    esi
12150 0001132F EBD2                   <1>       jmp    short pcmsg1
12151                                  <1> pcmsg5:
12152                                  <1>       ; 07/12/2020
12153 00011331 803D[DA6F0000]07       <1>         cmp     byte [CRT_MODE], 7
12154 00011338 76C9                   <1>       jna    short pcmsg1
12155                                  <1>
12156                                  <1>       ; 05/12/2020
12157                                  <1>       ; writing message by using
12158                                  <1>       ; VESA VBE3 video bios protected mode interface
12159                                  <1>
12160 0001133A B40E                   <1>       mov    ah, 0Eh
12161                                  <1> pcmsg6:
12162 0001133C AC                     <1>       lodsb
12163 0001133D 20C0                   <1>       and    al, al  ; 0
12164 0001133F 7407                   <1>       jz     short pcmsg2
12165                                  <1>       ; bh = video page
12166                                  <1>       ; ah = 0Eh
12167                                  <1>       ; al = character
12168                                  <1>       ; bl = color
12169 00011341 E8CC06FFFF             <1>       call   int10h_32bit_pmi
12170 00011346 EBF4                   <1>       jmp    short pcmsg6
12171                                  <1> pcmsg2:
12172 00011348 C3                     <1>       retn
12173                                  <1>
12174                                  <1> sysgeterr:
12175                                  <1>       ; 09/12/2015
12176                                  <1>       ; 21/09/2015 - (Retro UNIX 386 v1 feature only!)
12177                                  <1>       ; Get last error number or page fault count
12178                                  <1>       ; (for debugging)
12179                                  <1>       ;
12180                                  <1>       ; Input -> EBX = return type
12181                                  <1>       ;         0 = last error code (which is in 'u.error')
12182                                  <1>       ;         FFFFFFFFh = page fault count for running process
12183                                  <1>       ;         FFFFFFFEh = total page fault count
12184                                  <1>       ;         1 .. FFFFFFFDh = undefined
12185                                  <1>       ;
12186                                  <1>       ; Output -> EAX = last error number or page fault count
12187                                  <1>       ;         (depending on EBX input)
12188                                  <1>       ;
12189 00011349 21DB                   <1>       and    ebx, ebx
12190 0001134B 750B                   <1>       jnz    short glerr_2
12191                                  <1> glerr_0:
12192 0001134D A1[C8030300]           <1>       mov    eax, [u.error]
12193                                  <1> glerr_1:
12194 00011352 A3[64030300]           <1>       mov    [u.r0], eax
12195 00011357 C3                     <1>       retn
12196                                  <1> glerr_2:
12197 00011358 43                     <1>       inc    ebx ; FFFFFFFFh -> 0, FFFFFFFEh -> FFFFFFFFh
12198 00011359 74FD                   <1>       jz     short glerr_2 ; page fault count for process
12199 0001135B 43                     <1>       inc    ebx ; FFFFFFFFh -> 0
12200 0001135C 75EF                   <1>       jnz    short glerr_0
12201 0001135E A1[80050300]           <1>       mov    eax, [PF_Count] ; total page fault count
12202 00011363 EBED                   <1>         jmp     short glerr_1
12203                                  <1> glerr_3:
12204 00011365 A1[CC030300]           <1>       mov    eax, [u.pfcount]
12205 0001136A EBE6                   <1>       jmp    short glerr_1
12206                                  <1>
12207                                  <1> load_and_run_file:
12208                                  <1>       ; 18/11/2017
12209                                  <1>       ; 22/01/2017
12210                                  <1>       ; 04/01/2017, 07/01/2017
```

```
12211                                  <1>       ; 24/10/2016
12212                                  <1>       ; 24/04/2016, 02/05/2016, 03/05/2016, 06/05/2016
12213                                  <1>       ; 23/04/2016 (TRDOS 386 = TRDOS v2.0)
12214                                  <1>       ; 23/10/2015 (Retro UNIX 386 v1, 'sysexec')
12215                                  <1>       ; 23/06/2015 (Retro UNIX 386 v1 - Beginning)
12216                                  <1>       ; 03/06/2013 - 06/12/2013 (Retro UNIX 8086 v1)
12217                                  <1>       ; EAX = First Cluster number
12218                                  <1>       ; EDX = File Size
12219                                  <1>       ; ESI = Argument list address
12220                                  <1>       ; [argc] = argument count
12221                                  <1>       ; [u.nread] = argument list length
12222                                  <1>       ; [esp] = return address to the caller (*)
12223                                  <1>       ;
12224 0001136C 8935[4C040300]          <1>       mov    [argv], esi
12225 00011372 8915[55040300]          <1>       mov    [i.size], edx
12226 00011378 A3[51040300]            <1>       mov    [ii], eax
12227                                  <1>
12228                                  <1>       ;sti   ; 07/01/2017
12229                                  <1>       ;mov   eax, [k_page_dir]
12230                                  <1>       ;mov   [u.pgdir], eax
12231 0001137D 31C0                    <1>       xor    eax, eax ; clc ; *** ; 04/01/2017
12232                                  <1>       ;mov   [u.r0], eax ; 0 ; 07/01/2017
12233                                  <1>
12234                                  <1>       ; 06/05/2016
12235                                  <1>       ; Set 'sysexit' return order to MainProg
12236                                  <1>       ;
12237 0001137F 58                     <1>       pop    eax ; * 'loc_load_and_run_file_8:' address
12238                                  <1>       ;; 22/01/2017
12239                                  <1>       ;;cli ; 07/01/2017
12240 00011380 8B25[B4890100]          <1>       mov    esp, [tss.esp0]
12241                                  <1>       ;
12242                                  <1>       ; 'loc_load_run_file_8' address has
12243                                  <1>       ; 'jmp loc_file_rw_restore_retn' instruction
12244                                  <1>       ; 'loc_file_rw_restore_retn:' will return to
12245                                  <1>       ; [mainprog_return_addr]
12246                                  <1>       ; just after 'call command_interpreter'
12247                                  <1>       ;
12248 00011386 68[43750000]            <1>       push   _end_of_mainprog ; we must not return to here !
12249 0001138B FF35[98960100]          <1>       push   dword [mainprog_return_addr]
12250 00011391 89E5                    <1>       mov    ebp, esp ; **
12251                                  <1>       ;
12252 00011393 9C                      <1>       pushfd  ; EFLAGS      ; IRETD ; ***
12253 00011394 6A08                    <1>       push   KCODE ; cs     ; IRETD
12254 00011396 50                      <1>       push   eax ; * (eip) ; IRETD
12255 00011397 8925[5C030300]          <1>       mov    [u.sp], esp
12256                                  <1>       ;mov   byte [u.quant], time_count
12257 0001139D 1E                      <1>       push   ds
12258 0001139E 06                      <1>       push   es
12259 0001139F 0FA0                    <1>       push   fs
12260 000113A1 0FA8                    <1>       push   gs
12261                                  <1>       ;mov   eax, [u.r0]
12262 000113A3 29C0                    <1>       sub    eax, eax
12263 000113A5 60                      <1>       pushad
12264 000113A6 68[0CD90000]            <1>       push   sysret
12265                                  <1>       ;push  sysrel1 ; 07/01/2017
12266 000113AB 8925[60030300]          <1>       mov    [u.usp], esp
12267                                  <1>       ;
12268 000113B1 E845060000              <1>       call   wswap ; Save MainProg (process 1) 'u' structure
12269                                  <1>                    ; and registers for return (from program)
12270 000113B6 89EC                    <1>       mov    esp, ebp ; **
12271                                  <1>       ;;22/01/2017
12272                                  <1>       ;;sti ; 07/01/2017
12273 000113B8 50                      <1>       push   eax  ; * 'loc_load_and_run_file_8:' address
12274                                  <1>       ;
12275                                  <1>       ;;; 02/05/2016
12276                                  <1>       ;;; Create a new process (parent: MainProg)
12277 000113B9 31F6                    <1>       xor    esi, esi
12278                                  <1> cnpm_1: ; search p.stat table for unused process number
12279 000113BB 46                      <1>       inc    esi
12280 000113BC 80BE[AF000300]00        <1>       cmp    byte [esi+p.stat-1], 0 ; SFREE
12281                                  <1>                           ; is process active, unused, dead
12282 000113C3 760B                    <1>       jna    short cnpm_2 ; it's unused so branch
12283 000113C5 6683FE10                <1>       cmp    si, nproc    ; all processes checked
12284 000113C9 72F0                    <1>       jb     short cnpm_1    ; no, branch back
12285 000113CB E9FB61FFFF              <1>       jmp    panic
12286                                  <1> cnpm_2:
12287 000113D0 A1[B8030300]            <1>       mov    eax, [u.pgdir] ; page directory of MainProg
12288 000113D5 A3[BC030300]            <1>       mov    [u.ppgdir], eax ; parent's page directory
12289 000113DA E8FC47FFFF              <1>       call   allocate_page
12290 000113DF 0F82E661FFFF            <1>       jc     panic
12291                                  <1>       ; EAX = UPAGE (user structure page) address
12292 000113E5 A3[B4030300]            <1>       mov    [u.upage], eax ; memory page for 'user' struct (child)
12293 000113EA 89F7                    <1>       mov    edi, esi
12294 000113EC 66C1E702                <1>       shl    di, 2
12295 000113F0 8987[BC000300]          <1>       mov    [edi+p.upage-4], eax ; memory page for 'user' struct
12296 000113F6 E85A48FFFF              <1>       call   clear_page ; 03/05/2016
12297                                  <1>       ;movzx eax, byte [p.ttyc] ; console tty (for MainProg)
12298 000113FB 6629C0                  <1>       sub    ax, ax ; 0
12299 000113FE 668986[7F000300]        <1>       mov     [esi+p.ttyc-1], ax ; al - set child's console tty
12300                                  <1>                                ; ah - reset child's wait channel
12301 00011405 89F0                    <1>       mov    eax, esi
12302 00011407 A2[B3030300]            <1>       mov    [u.uno], al ; child process number
12303 0001140C FE86[AF000300]          <1>         inc     byte [esi+p.stat-1] ; 1, SRUN
12304 00011412 66D1E6                  <1>       shl    si, 1 ; multiply si by 2 to get index into p.pid table
12305 00011415 66FF05[4E030300]        <1>       inc    word [mpid] ; increment m.pid; get a new process name
12306 0001141C 66A1[4E030300]          <1>       mov    ax, [mpid]
12307 00011422 668986[1E000300]        <1>       mov    [esi+p.pid-2], ax ; put new process name
12308                                  <1>                              ; in child process' name slot
12309                                  <1>       ;mov   ax, [p.pid]  ; get process name of MainProg
12310 00011429 66B80100                <1>       mov    ax, 1
12311 0001142D 668986[3E000300]        <1>       mov    [esi+p.ppid-2], ax ; put parent process name
12312                                  <1>                                 ; in parent process slot for child
12313 00011434 6648                    <1>       dec    ax ; 0
12314 00011436 66A3[94030300]          <1>       mov    [u.ttyp], ax ; 0
12315                                  <1>       ;;;
```

```
12316 0001143C A1[51040300]        <1>       mov    eax,  [ii]
12317                              <1>       ; Retro UNIX 386 v1, 'sysexec' (u2.s)
12318 00011441 E84A170000          <1>       call   iopen
12319                              <1>       ; 06/06/2016
12320 00011446 C605[A9030300]01    <1>       mov    byte [u.pri], 1 ; normal priority
12321                              <1>       ;
12322 0001144D EB16                <1>       jmp    short sysexec_7 ; 02/05/2016
12323                              <1>
12324                              <1> sysexec_6:
12325                              <1>       ; 19/11/2017
12326                              <1>       ; 18/11/2017
12327                              <1>       ; 14/11/2017
12328                              <1>       ; 13/11/2017
12329 0001144F 8925[4C040300]      <1>       mov    [argv], esp ; *!* ; start address of argument list
12330                              <1>
12331                              <1>       ; 04/01/2017
12332                              <1>       ; 24/10/2016
12333                              <1>       ;;02/05/2016
12334                              <1>       ; 23/04/2016 (TRDOS 386)
12335                              <1>       ; 18/10/2015 ('sysexec_6')
12336                              <1>       ; 23/06/2015
12337 00011455 A1[B8030300]        <1>       mov    eax, [u.pgdir] ; physical address of page directory
12338                              <1>       ;cmp   eax, [k_page_dir] ; TRDOS MainProg ?
12339                              <1>       ;je    short sysexec_7
12340                              <1>       ; 19/11/2017
12341 0001145A 8B1D[BC030300]      <1>       mov    ebx, [u.ppgdir] ; phy addr of the parent's page dir
12342 00011460 E8AF48FFFF          <1>       call   deallocate_page_dir
12343                              <1> sysexec_7:
12344 00011465 E8DF47FFFF          <1>       call   make_page_dir
12345 0001146A 0F825B61FFFF        <1>       jc     panic  ; allocation error
12346                              <1>                     ; after a deallocation would be nonsence !?
12347                              <1>       ; 24/07/2015
12348                              <1>       ; map kernel pages (1st 4MB) to PDE 0
12349                              <1>       ;     of the user's page directory
12350                              <1>       ;     (It is needed for interrupts!)
12351                              <1>       ; 18/10/2015
12352 00011470 8B15[188A0100]      <1>       mov    edx, [k_page_dir] ; Kernel's page directory
12353 00011476 8B02                <1>       mov    eax, [edx] ; physical address of
12354                              <1>                       ; kernel's first page table (1st 4 MB)
12355                              <1>                       ; (PDE 0 of kernel's page directory)
12356 00011478 8B15[B8030300]      <1>       mov    edx, [u.pgdir]
12357 0001147E 8902                <1>       mov    [edx], eax ; PDE 0 (1st 4MB)
12358                              <1>       ;
12359                              <1>       ; 20/07/2015
12360 00011480 BB00004000          <1>       mov    ebx, CORE ; start address = 0 (virtual) + CORE
12361                              <1>       ; 18/10/2015
12362 00011485 BE[3C040300]        <1>       mov    esi, pcore ; physical start address
12363                              <1> sysexec_8:
12364 0001148A B907000000          <1>       mov    ecx, PDE_A_USER + PDE_A_WRITE + PDE_A_PRESENT
12365 0001148F E8D347FFFF          <1>       call   make_page_table
12366 00011494 0F823161FFFF        <1>       jc     panic
12367                              <1>       ;mov   ecx, PTE_A_USER + PTE_A_WRITE + PTE_A_PRESENT
12368 0001149A E8D647FFFF          <1>       call   make_page ; make new page, clear and set the pte
12369 0001149F 0F822661FFFF        <1>       jc     panic
12370                              <1>       ;
12371 000114A5 8906                <1>       mov    [esi], eax ; 24/06/2015
12372                              <1>       ; ebx = virtual address (24/07/2015)
12373 000114A7 E86E4DFFFF          <1>       call   add_to_swap_queue
12374                              <1>       ; 18/10/2015
12375 000114AC 81FE[40040300]      <1>       cmp    esi, ecore ; user's stack (last) page ?
12376 000114B2 740C                <1>       je     short sysexec_9 ; yes
12377 000114B4 BE[40040300]        <1>       mov    esi, ecore  ; physical address of the last page
12378                              <1>       ; 20/07/2015
12379 000114B9 BB00F0FFFF          <1>       mov    ebx, (ECORE - PAGE_SIZE) + CORE
12380                              <1>       ; ebx = virtual end address + segment base address - 4K
12381 000114BE EBCA                <1>         jmp     short sysexec_8
12382                              <1> sysexec_9:
12383                              <1>       ; 19/11/2017
12384                              <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
12385                              <1>       ; 25/06/2015, 26/08/2015, 18/10/2015
12386                              <1>       ; move arguments from kernel stack to [ecore]
12387                              <1>       ; (argument list/line will be copied from kernel stack
12388                              <1>       ; frame to the last (stack) page of user's core memory)
12389                              <1>       ; 18/10/2015
12390 000114C0 8B3D[40040300]      <1>       mov    edi, [ecore]
12391 000114C6 81C700100000        <1>       add    edi, PAGE_SIZE
12392                              <1>       ; 19/11/2017
12393 000114CC 83EF04              <1>       sub    edi, 4
12394 000114CF C70700000000        <1>       mov    dword [edi], 0
12395 000114D5 89FB                <1>       mov    ebx, edi
12396                              <1>       ;
12397 000114D7 0FB705[4A040300]    <1>       movzx  eax, word [argc]
12398 000114DE 09C0                <1>       or     eax, eax
12399 000114E0 7445                <1>       jz     short sysexec_13 ; 19/11/2017
12400                              <1>       ;jnz   short sysexec_10
12401                              <1>       ;mov   ebx, edi
12402                              <1>       ;sub   ebx, 4
12403                              <1>       ;mov   [ebx], eax ; 0
12404                              <1>       ;jmp   short sysexec_13
12405                              <1> sysexec_10:
12406 000114E2 8B0D[8C030300]      <1>       mov    ecx, [u.nread]
12407                              <1>       ; 13/11/2017
12408                              <1>       ;mov   esi, TextBuffer ; 'load_and_execute_file'
12409                              <1>       ;mov   esi, esp     ; 'sysexec'
12410 000114E8 8B35[4C040300]      <1>       mov    esi, [argv] ; 24/04/2016 (TRDOS 386  = TRDOS v2.0)
12411                              <1>       ;sub   edi, ecx ; page end address - argument list length
12412 000114EE 29CB                <1>       sub    ebx, ecx ; 19/11/2017
12413 000114F0 89C2                <1>       mov    edx, eax
12414 000114F2 FEC2                <1>       inc    dl ; argument count + 1 for argc value
12415 000114F4 C0E202              <1>       shl    dl, 2  ; 4 * (argument count + 1)
12416                              <1>       ;mov   ebx, edi
12417 000114F7 89DF                <1>       mov    edi, ebx ; 19//11/2017
12418 000114F9 80E3FC              <1>       and    bl, 0FCh ; 32 bit (dword) alignment
12419 000114FC 29D3                <1>       sub    ebx, edx
12420 000114FE 89FA                <1>       mov    edx, edi
```

```
12421 00011500 F3A4               <1>       rep    movsb
12422 00011502 89D6               <1>       mov    esi, edx
12423 00011504 89DF               <1>       mov    edi, ebx
12424 00011506 BA00F0BFFF         <1>       mov    edx, ECORE - PAGE_SIZE ; virtual addr. of the last page
12425 0001150B 2B15[40040300]     <1>       sub    edx, [ecore] ; difference (virtual - physical)
12426 00011511 AB                 <1>       stosd  ; eax = argument count
12427                             <1> sysexec_11:
12428 00011512 89F0               <1>       mov    eax, esi
12429 00011514 01D0               <1>       add    eax, edx
12430 00011516 AB                 <1>       stosd  ; eax = virtual address
12431                             <1>       ;dec   byte [argc]
12432 00011517 66FF0D[4A040300]   <1>       dec    word [argc] ; 14/11/2017
12433 0001151E 7407               <1>       jz     short sysexec_13
12434                             <1> sysexec_12:
12435 00011520 AC                 <1>       lodsb
12436 00011521 20C0               <1>       and    al, al
12437 00011523 75FB               <1>       jnz    short sysexec_12
12438 00011525 EBEB               <1>       jmp    short sysexec_11
12439                             <1> sysexec_13:
12440                             <1>       ; 24/10/2016
12441                             <1>       ; 24/04/2016 - TRDOS 386 (TRDOS v2.0)
12442                             <1>       ; 23/06/2015 - 19/10/2015 (Retro UNIX 386 v1, 'sysexec_13')
12443                             <1>       ;
12444                             <1>       ; moving arguments to [ecore] is OK here..
12445                             <1>       ;
12446                             <1>       ; ebx = beginning addres of argument list pointers
12447                             <1>             ;      in user's stack
12448 00011527 2B1D[40040300]     <1>       sub    ebx, [ecore]
12449 0001152D 81C300F0BFFF       <1>       add     ebx, (ECORE - PAGE_SIZE)
12450                             <1>                   ; end of core - 4096 (last page)
12451                             <1>                   ; (virtual address)
12452 00011533 891D[4C040300]     <1>       mov    [argv], ebx
12453 00011539 891D[90030300]     <1>       mov    [u.break], ebx ; available user memory
12454                             <1>       ;
12455 0001153F 29C0               <1>       sub    eax, eax
12456 00011541 C705[88030300]2000- <1>      mov    dword [u.count], 32 ; Executable file header size
12456 00011549 0000               <1>
12457 0001154B C705[74030300]-    <1>       mov    dword [u.fofp], u.off
12457 00011551 [80030300]         <1>
12458 00011555 A3[80030300]       <1>       mov    [u.off], eax ; 0
12459 0001155A A3[84030300]       <1>       mov    [u.base], eax ; 0, start of user's core (virtual)
12460                             <1>       ; 24/10/2016
12461 0001155F A0[DE8A0100]       <1>       mov    al, [Current_Drv]
12462 00011564 A2[46030300]       <1>       mov    [cdev], al
12463                             <1>       ;
12464 00011569 A1[51040300]       <1>       mov    eax, [ii] ; Fist Cluster of the Program (PRG) file
12465                             <1>       ; EAX = First cluster of the executable file
12466 0001156E E80A010000         <1>       call   readi
12467                             <1>
12468 00011573 8B0D[90030300]     <1>       mov    ecx, [u.break] ; top of user's stack (physical addr.)
12469 00011579 890D[88030300]     <1>       mov    [u.count], ecx ; save for overrun check
12470                             <1>       ;
12471 0001157F 8B0D[8C030300]     <1>       mov    ecx, [u.nread]
12472 00011585 890D[90030300]     <1>       mov    [u.break], ecx ; virtual address (offset from start)
12473 0001158B 80F920             <1>       cmp    cl, 32
12474 0001158E 7540               <1>         jne     short sysexec_15
12475                             <1>       ::
12476                             <1>       ; Retro UNIX 386 v1 (32 bit) executable file header format
12477 00011590 8B35[3C040300]     <1>       mov    esi, [pcore] ; start address of user's core memory
12478                             <1>                           ; (phys. start addr. of the exec. file)
12479 00011596 AD                 <1>       lodsd
12480 00011597 663DEB1E           <1>       cmp    ax, 1EEBh ; EBH, 1Eh -> jump to +32
12481 0001159B 7533               <1>       jne    short sysexec_15
12482 0001159D AD                 <1>       lodsd
12483 0001159E 89C1               <1>       mov    ecx, eax ; text (code) section size
12484 000115A0 AD                 <1>       lodsd
12485 000115A1 01C1               <1>       add    ecx, eax ; + data section size (initialized data)
12486 000115A3 89CB               <1>       mov    ebx, ecx
12487 000115A5 AD                 <1>       lodsd
12488 000115A6 01C3               <1>       add    ebx, eax ; + bss section size (for overrun checking)
12489 000115A8 3B1D[88030300]     <1>       cmp    ebx, [u.count]
12490 000115AE 7711               <1>       ja     short sysexec_14  ; program overruns stack !
12491                             <1>       ;
12492                             <1>       ; add bss section size to [u.break]
12493 000115B0 0105[90030300]     <1>       add    [u.break], eax
12494                             <1>       ;
12495 000115B6 83E920             <1>       sub    ecx, 32  ; header size (already loaded)
12496                             <1>       ;cmp   ecx, [u.count]
12497                             <1>       ;jnb   short sysexec_16
12498 000115B9 890D[88030300]     <1>       mov    [u.count], ecx ; required read count
12499 000115BF EB29               <1>       jmp    short sysexec_16
12500                             <1> sysexec_14:
12501                             <1>       ; insufficient (out of) memory
12502 000115C1 C705[C8030300]0400- <1>      mov    dword [u.error], ERR_MINOR_IM ; 1
12502 000115C9 0000               <1>
12503 000115CB E91CC3FFFF         <1>       jmp    error
12504                             <1> sysexec_15:
12505 000115D0 8B15[55040300]     <1>         mov  edx, [i.size] ; file size
12506 000115D6 29CA               <1>       sub    edx, ecx ; file size - loaded bytes
12507 000115D8 7626               <1>       jna    short sysexec_17 ; no need to next read
12508 000115DA 01D1               <1>       add    ecx, edx ; [i.size]
12509 000115DC 3B0D[88030300]     <1>       cmp    ecx, [u.count] ; overrun check (!)
12510 000115E2 77DD               <1>       ja     short sysexec_14
12511 000115E4 8915[88030300]     <1>       mov    [u.count], edx
12512                             <1> sysexec_16:
12513 000115EA A1[51040300]       <1>       mov    eax, [ii] ; first cluster
12514 000115EF E889000000         <1>       call   readi
12515 000115F4 8B0D[8C030300]     <1>       mov    ecx, [u.nread]
12516 000115FA 010D[90030300]     <1>       add    [u.break], ecx
12517                             <1> sysexec_17:
12518 00011600 A1[51040300]       <1>       mov    eax, [ii] ; first cluster
12519 00011605 E886150000         <1>       call   iclose
12520 0001160A 31C0               <1>       xor     eax, eax
12521 0001160C FEC0               <1>       inc    al
12522 0001160E 66A3[AA030300]     <1>       mov    [u.intr], ax ; 1 (interrupt/time-out is enabled)
```

```
12523 00011614 66A3[AC030300]      <1>       mov    [u.quit], ax ; 1 ('crtl+brk' signal is enabled)
12524 0001161A 833D[BC030300]00    <1>         cmp  dword [u.ppgdir], 0  ; is the caller MainProg (kernel) ?
12525 00011621 770C                <1>       ja     short sysexec_18 ; no, the caller is user process
12526                              <1>       ; If the caller is kernel (MainProg), 'sysexec' will come here
12527 00011623 8B15[188A0100]      <1>       mov    edx, [k_page_dir] ; kernel's page directory
12528 00011629 8915[BC030300]      <1>       mov    [u.ppgdir], edx ; next time 'sysexec' must not come here
12529                              <1> sysexec_18:
12530                              <1>       ; 02/05/2016
12531                              <1>       ; 24/04/2016 (TRDOS 386 = TRDOS v2.0)
12532                              <1>       ; 18/10/2015 (Retro UNIX 386 v1)
12533                              <1>       ; 05/08/2015
12534                              <1>       ; 29/07/2015
12535                              <1>
12536                              <1> ;     ; **** arguments list test start - 19/11/2017
12537                              <1> ;     mov    ebp, [argv]
12538                              <1> ;     sub    ebp, ECORE - 4096
12539                              <1> ;     add    ebp, [ecore]
12540                              <1> ;
12541                              <1> ;     mov    ebx, [ebp]
12542                              <1> ;     mov    [argc], bx
12543                              <1> ;     add    ebp, 4
12544                              <1> ;     mov    byte [ccolor], 1Fh
12545                              <1> ;_zx0:
12546                              <1> ;     cmp    word [argc], 0
12547                              <1> ;     jna    short _zx2
12548                              <1> ;_zx1:
12549                              <1> ;     push   ebp
12550                              <1> ;     mov    esi, [ebp]
12551                              <1> ;
12552                              <1> ;     sub    esi, ECORE - 4096
12553                              <1> ;     add    esi, [ecore]
12554                              <1> ;
12555                              <1> ;     call   print_cmsg
12556                              <1> ;
12557                              <1> ;     dec    word [argc]
12558                              <1> ;     jz     short _zx2
12559                              <1> ;
12560                              <1> ;     mov    al, '.'
12561                              <1> ;     mov    bl, 07h
12562                              <1> ;     mov    bh, [u.ttyn]
12563                              <1> ;     call   _write_tty
12564                              <1> ;
12565                              <1> ;     pop    ebp
12566                              <1> ;     add    ebp, 4
12567                              <1> ;     jmp    short _zx1
12568                              <1> ;_zx2:
12569                              <1> ;     pop    ebp
12570                              <1> ;     mov    byte [ccolor], 07h
12571                              <1> ;     mov    eax, 1
12572                              <1> ;     ; **** arguments list test stop
12573                              <1> ;     Test result is OK! (there is not a wrong thing) - 19/11/2017
12574                              <1>
12575 0001162F 8B2D[4C040300]      <1>       mov    ebp, [argv] ; user's stack pointer must point to argument
12576                              <1>                           ; list pointers (argument count)
12577 00011635 FA                  <1>       cli
12578 00011636 8B25[B4890100]      <1>         mov     esp, [tss.esp0] ; ring 0 (kernel) stack pointer
12579                              <1>       ;mov         esp, [u.sp] ; Restore Kernel stack
12580                              <1>                         ; for this process
12581                              <1>       ;add   esp, 20 ; --> EIP, CS, EFLAGS, ESP, SS
12582                              <1>       ;xor   eax, eax ; 0
12583 0001163C FEC8                <1>       dec    al ; eax = 0
12584                              <1>       ;mov   edx, UDATA
12585                              <1>       ; 18/11/2017
12586 0001163E 6A23                <1>       push   UDATA ; user's stack segment
12587                              <1>       ;push  edx
12588 00011640 55                  <1>       push   ebp ; user's stack pointer
12589                              <1>                  ; (points to number of arguments)
12590                              <1>
12591                              <1>       ; 04/01/2017
12592                              <1>       ; MainProg comes here while [sysflg]= 0FFh
12593                              <1>       ; (but sysexec comes here while [sysflg]= 0)
12594 00011641 C605[5B030300]00    <1>       mov    byte [sysflg], 0 ; 04/01/2017
12595                              <1>                               ; (timer_int sysflg control)
12596 00011648 FB                  <1>       sti
12597 00011649 9C                  <1>       pushfd ; EFLAGS
12598                              <1>              ; Set IF for enabling interrupts in user mode
12599                              <1>       ;or    dword [esp], 200h
12600                              <1>       ;
12601                              <1>       ;mov   bx, UCODE
12602                              <1>       ;push  bx ; user's code segment
12603 0001164A 6A1B                <1>       push   UCODE
12604                              <1>       ;push  0
12605 0001164C 50                  <1>       push   eax ; EIP (=0) - start address -
12606 0001164D 8925[5C030300]      <1>       mov    [u.sp], esp ; 29/07/2015
12607                              <1>       ; 05/08/2015
12608                              <1>       ; Remedy of a General Protection Fault during 'iretd' is here !
12609                              <1>       ; ('push dx' would cause to general protection fault,
12610                              <1>       ; after 'pop ds' etc.)
12611                              <1>       ;
12612                              <1>       ;; push dx ; ds (UDATA)
12613                              <1>       ;; push dx ; es (UDATA)
12614                              <1>       ;; push dx ; fs (UDATA)
12615                              <1>       ;; push dx ; gs (UDATA)
12616                              <1>       ;
12617                              <1>       ; This is a trick to prevent general protection fault
12618                              <1>       ; during 'iretd' intruction at the end of 'sysrele' (in u1.s):
12619 00011653 66BA2300            <1>       mov    dx, UDATA ; 19/11/2017
12620 00011657 8EC2                <1>       mov    es, dx ; UDATA
12621 00011659 06                  <1>       push   es ; ds (UDATA)
12622 0001165A 06                  <1>       push   es ; es (UDATA)
12623 0001165B 06                  <1>       push   es ; fs (UDATA)
12624 0001165C 06                  <1>       push   es ; gs (UDATA)
12625 0001165D 66BA1000            <1>       mov    dx, KDATA
12626 00011661 8EC2                <1>       mov    es, dx
12627                              <1>       ;
```

```
12628                                  <1>       ;; pushad simulation
12629 00011663 89E5                    <1>       mov    ebp, esp ; esp before pushad
12630 00011665 50                      <1>       push   eax ; eax (0)
12631 00011666 50                      <1>       push   eax ; ecx (0)
12632 00011667 50                      <1>       push   eax ; edx (0)
12633 00011668 50                      <1>       push   eax ; ebx (0)
12634 00011669 55                      <1>       push   ebp ; esp before pushad
12635 0001166A 50                      <1>       push   eax ; ebp (0)
12636 0001166B 50                      <1>       push   eax ; esi (0)
12637 0001166C 50                      <1>       push   eax ; edi (0)
12638                                  <1>       ;
12639 0001166D A3[64030300]            <1>       mov    [u.r0], eax ; eax = 0
12640 00011672 8925[60030300]          <1>       mov    [u.usp], esp
12641                                  <1>
12642                                  <1>       ; 14/11/2017
12643 00011678 E991C2FFFF              <1>       jmp    sysret0
12644                                  <1>
12645                                  <1> ;     ; 02/05/2016
12646                                  <1> ;     ;inc   byte [sysflg] ; 0FFh -> 0
12647                                  <1> ;     ;mov   byte [sysflg], 0 ; 04/01/2017
12648                                  <1> ;     movzx  ebx, byte [u.uno]
12649                                  <1> ;     shl    bl, 1 ; 13/11/2017
12650                                  <1> ;     cmp    word [ebx+p.ppid-2], 1 ; MainProg
12651                                  <1> ;     ja     sysret0 ; 03/05/2016
12652                                  <1> ;     push   sysret ; *
12653                                  <1> ;     mov    [u.usp], esp
12654                                  <1> ;     call   wswap ; save child process 'u' structure and
12655                                  <1> ;                  ; registers
12656                                  <1> ;     add    dword [u.usp], 4 ; 03/05/2016
12657                                  <1> ;sysexec_19: ; 02/05/2016
12658                                  <1> ;     retn ; * 'sysret' ; byte [sysflg] -> 0FFh
12659                                  <1>
12660                                  <1> readi:
12661                                  <1>       ; 01/05/2016
12662                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
12663                                  <1>       ; 20/05/2015 - Retro UNIX 386 v1
12664                                  <1>       ; 11/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
12665                                  <1>       ;
12666                                  <1>       ; Reads from a file whose the first cluster number in EAX
12667                                  <1>       ;
12668                                  <1>       ; INPUTS ->
12669                                  <1>       ;    EAX - First cluster number of the file
12670                                  <1>       ;    u.count - byte count user desires
12671                                  <1>       ;    u.base - points to user buffer
12672                                  <1>       ;    u.fofp - points to dword with current file offset
12673                                  <1>       ;    i.size - file size
12674                                  <1>       ;    cdev - logical dos drive number of the file
12675                                  <1>       ; OUTPUTS ->
12676                                  <1>       ;    u.count - cleared
12677                                  <1>       ;    u.nread - accumulates total bytes passed back
12678                                  <1>       ;
12679                                  <1>       ; ((EAX)) input/output
12680                                  <1>       ; (Retro UNIX Prototype : 14/12/2012 - 01/03/2013, UNIXCOPY.ASM)
12681                                  <1>         ; ((Modified registers: edx, ebx, ecx, esi, edi))
12682                                  <1>
12683 0001167D 31D2                    <1>       xor    edx, edx ; 0
12684 0001167F 8915[8C030300]          <1>       mov    [u.nread], edx ; 0
12685 00011685 668915[C4030300]        <1>       mov    [u.pcount], dx ; 19/05/2015
12686 0001168C 3915[88030300]          <1>       cmp    [u.count], edx ; 0
12687 00011692 7701                    <1>       ja     short readi_1
12688 00011694 C3                      <1>       retn
12689                                  <1> readi_1:
12690                                  <1> dskr:
12691                                  <1>       ; 01/05/2016
12692                                  <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
12693                                  <1>       ; 24/05/2015 - 12/10/2015 (Retro UNIX 386 v1)
12694                                  <1>       ; 26/04/2013 - 03/08/2013 (Retro UNIX 8086 v1)
12695                                  <1> dskr_0:
12696 00011695 8B15[55040300]          <1>         mov edx, [i.size]
12697 0001169B 8B1D[74030300]          <1>       mov    ebx, [u.fofp]
12698 000116A1 2B13                    <1>       sub    edx, [ebx]
12699 000116A3 7647                    <1>       jna    short dskr_4
12700                                  <1>       ;
12701 000116A5 50                      <1>       push   eax ; 01/05/2016
12702 000116A6 3B15[88030300]          <1>       cmp     edx, [u.count]
12703 000116AC 7306                    <1>       jnb    short dskr_1
12704 000116AE 8915[88030300]          <1>       mov    [u.count], edx
12705                                  <1> dskr_1:
12706                                  <1>       ; EAX = First Cluster
12707                                  <1>       ; [Current_Drv] = Physical drive number
12708 000116B4 E83B000000              <1>       call   mget_r
12709                                  <1>       ; NOTE: in 'mget_r', relevant sector will be read in buffer
12710                                  <1>       ; if it is not already in buffer !
12711 000116B9 BB[8C050300]            <1>       mov    ebx, readi_buffer
12712 000116BE 803D[C6030300]00        <1>       cmp    byte [u.kcall], 0 ; the caller is 'namei' sign (=1)
12713 000116C5 770F                    <1>       ja     short dskr_3   ; zf=0 -> the caller is 'namei'
12714 000116C7 66833D[C4030300]00      <1>       cmp    word [u.pcount], 0
12715 000116CF 7705                    <1>       ja     short dskr_3
12716                                  <1> dskr_2:
12717                                  <1>       ; [u.base] = virtual address to transfer (as destination address)
12718 000116D1 E894010000              <1>       call   trans_addr_w ; translate virtual address to physical (w)
12719                                  <1> dskr_3:
12720                                  <1>       ; EBX (r5) = system (I/O) buffer address -physical-
12721 000116D6 E8F7010000              <1>       call   sioreg
12722 000116DB 87F7                    <1>       xchg   esi, edi
12723                                  <1>       ; EDI = file (user data) offset
12724                                  <1>       ; ESI = sector (I/O) buffer offset
12725                                  <1>       ; ECX = byte count
12726 000116DD F3A4                    <1>       rep    movsb
12727                                  <1>       ; eax = remain bytes in buffer
12728                                  <1>         ;       (check if remain bytes in the buffer > [u.pcount])
12729 000116DF 09C0                    <1>       or     eax, eax
12730 000116E1 75EE                    <1>       jnz    short dskr_2 ; (page end before system buffer end!)
12731 000116E3 58                      <1>       pop    eax  ; (first cluster number)
12732 000116E4 390D[88030300]          <1>       cmp    [u.count], ecx ; 0
```

```
12733 000116EA 77A9                <1>       ja     short dskr_0
12734                              <1> dskr_4:
12735 000116EC C605[C6030300]00    <1>       mov    byte [u.kcall], 0
12736 000116F3 C3                  <1>       retn
12737                              <1>
12738                              <1> mget_r:
12739                              <1>       ; 24/10/2016
12740                              <1>       ; 22/10/2016
12741                              <1>       ; 12/10/2016
12742                              <1>       ; 29/04/2016
12743                              <1>       ; 25/04/2016 - TRDOS 386 (TRDOS v2.0)
12744                              <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
12745                              <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
12746                              <1>       ;
12747                              <1>       ; Get existing or (allocate) a new disk block for file
12748                              <1>       ;
12749                              <1>       ; INPUTS ->
12750                              <1>       ;    [u.fofp] = file offset pointer
12751                              <1>       ;    EAX = First Cluster
12752                              <1>       ;    [cdev] = Logical dos drive number
12753                              <1>       ;    ([u.off] = file offset)
12754                              <1>       ; OUTPUTS ->
12755                              <1>       ;    EAX = logical sector number
12756                              <1>       ;    ESI = Logical Dos Drive Description Table address
12757                              <1>       ;
12758                              <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI
12759                              <1>
12760 000116F4 8B35[74030300]      <1>       mov     esi, [u.fofp]
12761 000116FA 8B1E                <1>       mov    ebx, [esi] ; (u.off)
12762                              <1>
12763 000116FC 29C9                <1>       sub    ecx, ecx
12764 000116FE 8A2D[46030300]      <1>       mov    ch, [cdev]
12765                              <1>
12766 00011704 BE00010900          <1>       mov    esi, Logical_DOSDisks
12767 00011709 01CE                <1>       add    esi, ecx
12768                              <1>
12769 0001170B 380D[4C960100]      <1>       cmp    [readi.valid], cl ; 0
12770 00011711 7649                <1>       jna    short mget_r_0
12771                              <1>
12772 00011713 3A2D[4D960100]      <1>       cmp    ch, [readi.drv]
12773 00011719 7541                <1>       jne    short mget_r_0
12774                              <1>
12775 0001171B 3B05[60960100]      <1>       cmp    eax, [readi.fclust]
12776 00011721 7565                <1>       jne    short mget_r_3
12777                              <1>
12778 00011723 89D8                <1>       mov    eax, ebx ; file offset
12779 00011725 668B0D[54960100]    <1>       mov    cx, [readi.bpc]
12780 0001172C 41                  <1>       inc    ecx ; <= 65536
12781 0001172D 29D2                <1>       sub    edx, edx
12782 0001172F F7F1                <1>       div    ecx
12783                              <1>
12784 00011731 8B3D[5C960100]      <1>       mov    edi, [readi.c_index] ; cluster index
12785                              <1>
12786 00011737 39F8                <1>       cmp    eax, edi
12787 00011739 757A                <1>         jne     short mget_r_4  ; (*)
12788                              <1>
12789                              <1>       ; edx = byte offset in cluster (<= 65535)
12790 0001173B 668915[56960100]    <1>       mov    [readi.offset], dx
12791 00011742 66C1EA09            <1>       shr    dx, 9 ; / 512
12792 00011746 8815[4F960100]      <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
12793                              <1>
12794 0001174C A1[58960100]        <1>       mov    eax, [readi.cluster]  ; > 0 if [readi.valid] = 1
12795 00011751 8B15[64960100]      <1>       mov    edx, [readi.fs_index]
12796 00011757 E99A000000          <1>         jmp     mget_r_7
12797                              <1>
12798                              <1> mget_r_0:
12799 0001175C 882D[4D960100]      <1>       mov    [readi.drv], ch ; physical drive number
12800 00011762 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
12801 00011766 7707                <1>       ja     short mget_r_1
12802 00011768 8A4E12              <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
12803 0001176B D0E9                <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
12804 0001176D EB03                <1>       jmp    short mget_r_2
12805                              <1> mget_r_1:
12806 0001176F 8A4E13              <1>       mov    cl, [esi+LD_BPB+BPB_SecPerClust]
12807                              <1> mget_r_2:
12808 00011772 880D[4E960100]      <1>       mov    [readi.spc], cl  ; sectors per cluster
12809                              <1>       ; NOTE: readi bytes per sector value is always 512 !
12810 00011778 66C1E109            <1>       shl    cx, 9 ; * 512
12811 0001177C 6649                <1>       dec    cx ; bytes per cluster - 1
12812 0001177E 66890D[54960100]    <1>       mov    [readi.bpc], cx
12813 00011785 6629C9              <1>       sub    cx, cx
12814                              <1> mget_r_3:
12815 00011788 A3[60960100]        <1>       mov    [readi.fclust], eax ; first cluster (or FDT address)
12816 0001178D 880D[4C960100]      <1>       mov    [readi.valid], cl ; 0
12817                              <1>       ;mov   [readi.s_index], cl ; 0
12818                              <1>       ;mov   [readi.offset], cx ; 0
12819 00011793 890D[5C960100]      <1>       mov    [readi.c_index], ecx ; 0
12820 00011799 890D[58960100]      <1>       mov    [readi.cluster], ecx ; 0
12821 0001179F 890D[50960100]      <1>       mov    [readi.sector], ecx ; 0
12822                              <1>
12823 000117A5 89D8                <1>       mov    eax, ebx ; file offset
12824 000117A7 668B0D[54960100]    <1>       mov    cx, [readi.bpc]
12825 000117AE 41                  <1>       inc    ecx ; <= 65536
12826 000117AF 29D2                <1>       sub    edx, edx
12827 000117B1 F7F1                <1>       div    ecx
12828                              <1>       ;mov   edi, [readi.c_index] ; previous cluster index
12829 000117B3 29FF                <1>       sub    edi, edi
12830                              <1> mget_r_4:
12831 000117B5 A3[5C960100]        <1>       mov    [readi.c_index], eax ; cluster index
12832                              <1>       ; edx = byte offset in cluster (<= 65535)
12833 000117BA 668915[56960100]    <1>       mov    [readi.offset], dx
12834 000117C1 66C1EA09            <1>       shr    dx, 9 ; / 512
12835 000117C5 8815[4F960100]      <1>       mov    [readi.s_index], dl ; sector index in cluster (0 to spc -1)
12836                              <1>
12837 000117CB 89C1                <1>       mov    ecx, eax ; current cluster index
```

```
12838 000117CD A1[60960100]       <1>       mov   eax, [readi.fclust]
12839 000117D2 09C9               <1>       or    ecx, ecx ; cluster index
12840 000117D4 741B               <1>       jz    short mget_r_6
12841                             <1>
12842 000117D6 39CF               <1>       cmp   edi, ecx
12843 000117D8 7710               <1>       ja    short mget_r_5 ; old cluster index is higher
12844 000117DA 8B15[58960100]     <1>       mov   edx, [readi.cluster]
12845 000117E0 21D2               <1>       and   edx, edx
12846 000117E2 7406               <1>       jz    short mget_r_5
12847                             <1>       ; valid 'readi' parameters (*)
12848 000117E4 89D0               <1>       mov   eax, edx
12849 000117E6 29F9               <1>       sub   ecx, edi
12850 000117E8 740C               <1>       jz    short mget_r_7
12851                             <1> mget_r_5:
12852                             <1>       ; EAX = Beginning cluster
12853                             <1>       ; EDX = Sector index in disk/file section
12854                             <1>       ;       (Only for SINGLIX file system!)
12855                             <1>       ; ECX = Cluster sequence number after the beginning cluster
12856                             <1>       ; ESI = Logical DOS Drive Description Table address
12857 000117EA E896BFFFFF         <1>       call  get_cluster_by_index
12858 000117EF 724E               <1>       jc    short mget_r_err
12859                             <1>       ; EAX = Cluster number
12860                             <1> mget_r_6:
12861 000117F1 A3[58960100]       <1>       mov   [readi.cluster], eax ; FDT number for Singlix File System
12862                             <1> mget_r_7:
12863 000117F6 807E0300           <1>       cmp   byte [esi+LD_FATType], 0
12864 000117FA 765F               <1>       jna   short mget_r_12
12865                             <1>
12866 000117FC 83E802             <1>       sub   eax, 2
12867 000117FF 0FB615[4E960100]   <1>       movzx edx, byte [readi.spc]
12868 00011806 F7E2               <1>       mul   edx
12869                             <1>
12870 00011808 034668             <1>       add   eax, [esi+LD_DATABegin]
12871 0001180B 8A15[4F960100]     <1>       mov   dl, [readi.s_index]
12872 00011811 01D0               <1>       add   eax, edx
12873                             <1> mget_r_8:
12874                             <1>       ; eax = logical sector number
12875 00011813 803D[4C960100]00   <1>       cmp   byte [readi.valid], 0
12876 0001181A 7608               <1>       jna   short mget_r_9
12877 0001181C 3B05[50960100]     <1>       cmp   eax, [readi.sector]
12878 00011822 7436               <1>       je    short mget_r_11 ; sector is already in 'readi' buffer
12879                             <1> mget_r_9:
12880 00011824 A3[50960100]       <1>       mov   [readi.sector], eax
12881 00011829 BB[8C050300]       <1>       mov   ebx, readi_buffer ; buffer address
12882 0001182E B901000000         <1>       mov   ecx, 1
12883                             <1>       ; 29/04/2016
12884                             <1>       ;xor  dl, dl
12885                             <1>
12886                             <1>       ; EAX = Logical sector number
12887                             <1>       ; ECX = Sector count
12888                             <1>       ; EBX = Buffer address
12889                             <1>       ; (EDX = 0)
12890                             <1>       ; ESI = Logical DOS drive description table address
12891                             <1>
12892 00011833 E868130000         <1>       call  disk_read
12893 00011838 7314               <1>       jnc   short mget_r_10
12894                             <1>
12895                             <1>       ; 22/10/2016 (15h -> 17)
12896 0001183A B811000000         <1>       mov   eax, 17 ; Drive not ready or read error !
12897                             <1> mget_r_err:
12898 0001183F A3[C8030300]       <1>       mov   [u.error], eax
12899                             <1>       ; 12/10/2016
12900 00011844 A3[64030300]       <1>       mov   [u.r0], eax
12901 00011849 E99EC0FFFF         <1>       jmp   error
12902                             <1> mget_r_10:
12903 0001184E C605[4C960100]01   <1>       mov   byte [readi.valid], 1 ; 24/10/2016
12904 00011855 A1[50960100]       <1>       mov   eax, [readi.sector]
12905                             <1> mget_r_11:
12906 0001185A C3                 <1>       retn
12907                             <1> mget_r_12:
12908                             <1>       ; EAX = FDT number
12909                             <1>       ; EDX = Sector index from FDT sector (0,1,2,3,4...)
12910 0001185B 40                 <1>       inc   eax ; the first data sector in FS disk section
12911 0001185C 8915[64960100]     <1>       mov   [readi.fs_index], edx
12912 00011862 01D0               <1>       add   eax, edx
12913 00011864 EBAD               <1>       jmp   short mget_r_8
12914                             <1>
12915                             <1> trans_addr_r:
12916                             <1>       ; 12/10/2016
12917                             <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
12918                             <1>       ; Translate virtual address to physical address
12919                             <1>       ; for reading from user's memory space
12920                             <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
12921                             <1>
12922 00011866 31D2               <1>       xor   edx, edx ; 0 (read access sign)
12923 00011868 EB04               <1>       jmp   short trans_addr_rw
12924                             <1>
12925                             <1> trans_addr_w:
12926                             <1>       ; 12/10/2016
12927                             <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
12928                             <1>       ; Translate virtual address to physical address
12929                             <1>       ; for writing to user's memory space
12930                             <1>       ; 04/06/2015 - 18/10/2015 (Retro UNIX 386 v1)
12931                             <1>
12932 0001186A 29D2               <1>       sub   edx, edx
12933 0001186C FEC2               <1>       inc   dl ; 1 (write access sign)
12934                             <1> trans_addr_rw:
12935 0001186E 50                 <1>       push  eax
12936 0001186F 53                 <1>       push  ebx
12937 00011870 52                 <1>       push  edx ; r/w sign (in DL)
12938                             <1>       ;
12939 00011871 8B1D[84030300]     <1>       mov   ebx, [u.base]
12940 00011877 E8744AFFFF         <1>       call  get_physical_addr ; get physical address
12941 0001187C 730F               <1>       jnc   short passc_0
12942 0001187E A3[C8030300]       <1>       mov   [u.error], eax
```

```
12943 00011883 A3[64030300]        <1>       mov   [u.r0], eax ; 12/10/2016
12944                              <1>       ;pop  edx
12945                              <1>       ;pop  ebx
12946                              <1>       ;pop  eax
12947 00011888 E95FC0FFFF          <1>       jmp   error
12948                              <1> passc_0:
12949 0001188D F6C202              <1>       test  dl, PTE_A_WRITE ; writable page
12950 00011890 5A                  <1>       pop   edx
12951 00011891 751C                <1>       jnz   short passc_1
12952                              <1>
12953 00011893 20D2                <1>       and   dl, dl
12954 00011895 7418                <1>       jz    short passc_1
12955                              <1>       ; read only (duplicated) page -must be copied to a new page-
12956                              <1>       ; EBX = linear address
12957 00011897 51                  <1>       push  ecx
12958 00011898 E8EC46FFFF          <1>       call  copy_page
12959 0001189D 59                  <1>       pop   ecx
12960 0001189E 721E                <1>       jc    short passc_2
12961 000118A0 50                  <1>       push  eax ; physical address of the new/allocated page
12962 000118A1 E87449FFFF          <1>       call  add_to_swap_queue
12963 000118A6 58                  <1>       pop   eax
12964 000118A7 81E3FF0F0000        <1>       and   ebx, PAGE_OFF ; 0FFFh
12965                              <1>       ;mov  ecx, PAGE_SIZE
12966                              <1>       ;sub  ecx, ebx
12967 000118AD 01D8                <1>       add   eax, ebx
12968                              <1> passc_1:
12969 000118AF A3[C0030300]        <1>       mov   [u.pbase], eax ; physical address
12970 000118B4 66890D[C4030300]    <1>       mov   [u.pcount], cx ; remain byte count in page (1-4096)
12971 000118BB 5B                  <1>       pop   ebx
12972 000118BC 58                  <1>       pop   eax
12973 000118BD C3                  <1>       retn
12974                              <1> passc_2:
12975 000118BE B804000000          <1>       mov   eax, ERR_MINOR_IM ; "Insufficient memory !" error
12976 000118C3 A3[64030300]        <1>       mov   [u.r0], eax ; 12/10/2016
12977 000118C8 A3[C8030300]        <1>       mov   dword [u.error], eax
12978                              <1>       ;pop  ebx
12979                              <1>       ;pop  eax
12980 000118CD E91AC0FFFF          <1>       jmp   error
12981                              <1>
12982                              <1> sioreg:
12983                              <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
12984                              <1>       ; 19/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
12985                              <1>       ; 12/03/2013 - 22/07/2013 (Retro UNIX 8086 v1)
12986                              <1>       ; INPUTS ->
12987                              <1>       ;     EBX = system buffer (data) address (r5)
12988                              <1>       ;     [u.fofp] = pointer to file offset pointer
12989                              <1>       ;     [u.base] = virtual address of the user buffer
12990                              <1>       ;     [u.pbase] = physical address of the user buffer
12991                              <1>       ;     [u.count] = byte count
12992                              <1>       ;     [u.pcount] = byte count within page frame
12993                              <1>       ; OUTPUTS ->
12994                              <1>       ;     ESI = user data offset (r1)
12995                              <1>       ;     EDI = system (I/O) buffer offset (r2)
12996                              <1>       ;     ECX = byte count (r3)
12997                              <1>       ;     EAX = remain bytes after byte count within page frame
12998                              <1>       ;      (If EAX > 0, transfer will continue from the next page)
12999                              <1>         ;
13000                              <1>       ; ((Modified registers:  EDX))
13001                              <1>
13002 000118D2 8B35[74030300]      <1>         mov     esi, [u.fofp]
13003 000118D8 8B3E                <1>         mov     edi, [esi]
13004 000118DA 89F9                <1>       mov   ecx, edi
13005 000118DC 81C900FEFFFF        <1>       or    ecx, 0FFFFFE00h
13006 000118E2 81E7FF010000        <1>       and   edi, 1FFh
13007 000118E8 01DF                <1>       add   edi, ebx ; EBX = system buffer (data) address
13008 000118EA F7D9                <1>       neg   ecx
13009 000118EC 3B0D[88030300]      <1>       cmp   ecx, [u.count]
13010 000118F2 7606                <1>       jna   short sioreg_0
13011 000118F4 8B0D[88030300]      <1>       mov   ecx, [u.count]
13012                              <1> sioreg_0:
13013 000118FA 803D[C6030300]00    <1>       cmp   byte [u.kcall], 0
13014 00011901 7613                <1>       jna   short sioreg_1
13015                              <1>         ; the caller is 'mkdir' or 'namei'
13016 00011903 A1[84030300]        <1>       mov   eax, [u.base]
13017 00011908 A3[C0030300]        <1>       mov   [u.pbase], eax ; physical address = virtual address
13018 0001190D 66890D[C4030300]    <1>       mov   word [u.pcount], cx ; remain bytes in buffer (1 sector)
13019 00011914 EB0B                <1>       jmp   short sioreg_2
13020                              <1> sioreg_1:
13021 00011916 0FB715[C4030300]    <1>       movzx  edx, word [u.pcount]
13022 0001191D 39D1                <1>       cmp   ecx, edx
13023 0001191F 772A                <1>       ja    short sioreg_4 ; transfer count > [u.pcount]
13024                              <1> sioreg_2: ; 2:
13025 00011921 31C0                <1>       xor   eax, eax
13026                              <1> sioreg_3:
13027 00011923 010D[8C030300]      <1>       add   [u.nread], ecx
13028 00011929 290D[88030300]      <1>       sub   [u.count], ecx
13029 0001192F 010D[84030300]      <1>       add   [u.base], ecx
13030 00011935 010E                <1>         add [esi], ecx
13031 00011937 8B35[C0030300]      <1>       mov   esi, [u.pbase]
13032 0001193D 66290D[C4030300]    <1>       sub   [u.pcount], cx
13033 00011944 010D[C0030300]      <1>       add   [u.pbase], ecx
13034 0001194A C3                  <1>         retn
13035                              <1> sioreg_4:
13036                              <1>       ; transfer count > [u.pcount]
13037                              <1>       ; (ecx > edx)
13038 0001194B 89C8                <1>       mov   eax, ecx
13039 0001194D 29D0                <1>       sub   eax, edx ; remain bytes for 1 sector (block) transfer
13040 0001194F 89D1                <1>       mov   ecx, edx ; current transfer count = [u.pcount]
13041 00011951 EBD0                <1>       jmp   short sioreg_3
13042                              <1>
13043                              <1> tswitch: ; Retro UNIX 386 v1
13044                              <1> tswap:
13045                              <1>       ; 16/01/2017
13046                              <1>       ; 21/05/2016 - TRDOS 386 (TRDOS v2.0)
13047                              <1>       ; 10/05/2015 - 01/09/2015 (Retro UNIX 386 v1)
```

```
13048                                  <1>       ; 14/04/2013 - 14/02/2014 (Retro UNIX 8086 v1)
13049                                  <1>       ; time out swap, called when a user times out.
13050                                  <1>       ; the user is put on the low priority queue.
13051                                  <1>       ; This is done by making a link from the last user
13052                                  <1>       ; on the low priority queue to him via a call to 'putlu'.
13053                                  <1>       ; then he is swapped out.
13054                                  <1>
13055                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 21/05/2016 **
13056                                  <1>       ;     * when a high priority (event) process will be stopped
13057                                  <1>       ;      (swapped out, swithched out/off), 'tswap/tswitch' will
13058                                  <1>       ;      not add it to a run queue.
13059                                  <1>       ;      /// What for: Process may be already in a run queue,
13060                                  <1>       ;      it is unspeficied state because process might be started
13061                                  <1>       ;      by a timer event which does not regard previous priority
13062                                  <1>       ;      level and run queue of the process (for fast executing!).
13063                                  <1>       ;      After the 'run for event', process will be sequenced
13064                                  <1>       ;      to run by it's actual run queue. ///
13065                                  <1>       ;
13066                                  <1>       ; Retro UNIX 386 v1 modification ->
13067                                  <1>       ;       swap (software task switch) is performed by changing
13068                                  <1>       ;      user's page directory (u.pgdir) instead of segment change
13069                                  <1>       ;      as in Retro UNIX 8086 v1.
13070                                  <1>       ;
13071                                  <1>       ; RETRO UNIX 8086 v1 modification ->
13072                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
13073                                  <1>       ;      according to 8086 cpu (x86 real mode) architecture.
13074                                  <1>       ;      pdp-11 was using 64KB uniform memory while IBM PC
13075                                  <1>       ;      compatibles was using 1MB segmented memory
13076                                  <1>       ;      in 8086/8088 times.
13077                                  <1>       ;
13078                                  <1>       ; INPUTS ->
13079                                  <1>       ;    u.uno - users process number
13080                                  <1>       ;    runq+4 - lowest priority queue
13081                                  <1>       ; OUTPUTS ->
13082                                  <1>       ;    r0 - users process number
13083                                  <1>       ;    r2 - lowest priority queue address
13084                                  <1>       ;
13085                                  <1>       ; ((AX = R0, BX = R2)) output
13086                                  <1>       ; ((Modified registers: EDX, EBX, ECX, ESI, EDI))
13087                                  <1>       ;
13088                                  <1>
13089                                  <1>       NOTE:
13090                                  <1>       ;* [u.pri] priority level is specified by run queue which is process
13091                                  <1>       ;  comes to run from.
13092                                  <1>       ;* Initial [u.pri] is 1 ('normal/regular') for programs
13093                                  <1>       ;  (which are launched by MainProg or 'sysexec'), it is changed
13094                                  <1>       ;  to 2 ('high') by timer event, if program uses 'systimer' system call.
13095                                  <1>       ;* Program (Process) also can change it's running priority
13096                                  <1>       ;  from 1 to 0 or up to 2 by using 'syspri' system call; but,
13097                                  <1>       ;  if program selects priority level 2 (high) for running, next time
13098                                  <1>       ;  it is reduced to 1 (normal/regular) because 'syspri' adds this
13099                                  <1>       ;  program to 'run for normal' queue while running duration is a bit
13100                                  <1>       ;  protected from swap/switch out immediate, behalf of other high
13101                                  <1>       ;  priority process in sequence. Program (with high priority) will not
13102                                  <1>       ;  be swapped/switched out (by timer event) before it's time quantum
13103                                  <1>       ;  will be elapsed, but, this will be temporary if program is not using
13104                                  <1>       ;  timer event function.
13105                                  <1>
13106                                  <1>       ;For example:
13107                                  <1>       ;If a process frequently gets a timer event, it runs at high priority
13108                                  <1>       ;level but when it returns from running it returns to actual run queue,
13109                                  <1>       ;not to 'run for event' queue again.
13110                                  <1>       ;'tswap' will not change the sequence at return/stop(swap out) stage.
13111                                  <1>       ;But if priority level not high (=2, 'run for event'), 'tswap/tswitch'
13112                                  <1>       ;will add the stopping process to relevant run queue according to
13113                                  <1>       ;[u.pri] priority level.
13114                                  <1>
13115                                  <1>       ; 16/01/2017
13116 00011953 BB[54030300]            <1>       mov    ebx, runq+2  ; 'runq_normal' ; normal/regular priority
13117                                  <1>       ; 21/05/2016
13118                                  <1>       ;cmp   byte [u.pri], 2     ; high priority (run for event) ?
13119                                  <1>       ;jnb   short swap
13120                                  <1>       ; 16/01/2017
13121                                  <1>       ; (Normal and also high/event priority processes will be added to
13122                                  <1>       ; normal priority run queue for ensuring circular running sequence!)
13123                                  <1>       ; (Timer interrupt or 'syspri' system call may change priority and run
13124                                  <1>       ; queue to high/event level.)
13125 00011958 803D[A9030300]00        <1>       cmp    byte [u.pri], 0
13126 0001195F 7702                    <1>       ja     short tswap_1; normal priority run queue
13127                                  <1>       ;
13128 00011961 43                      <1>       inc    ebx
13129 00011962 43                      <1>       inc    ebx          ; runq+4, 'runq_background', low priority
13130                                  <1> tswap_1:
13131 00011963 A0[B3030300]            <1>       mov    al, [u.uno]
13132                                  <1>                     ; movb u.uno,r1 / move users process number to r1
13133                                  <1>              ; mov  $runq+4,r2
13134                                  <1>                     ; / move lowest priority queue address to r2
13135                                  <1>              ; ebx = run queue
13136 00011968 E8FE000000              <1>       call   putlu
13137                                  <1>              ; jsr r0,putlu / create link from last user on Q to
13138                                  <1>                          ; / u.uno's user
13139                                  <1>
13140                                  <1> switch: ; Retro UNIX 386 v1
13141                                  <1> swap:
13142                                  <1>       ; 02/01/2017
13143                                  <1>       ; 21/05/2016
13144                                  <1>       ; 20/05/2016
13145                                  <1>       ; 02/05/2016
13146                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13147                                  <1>       ; 10/05/2015 - 02/09/2015 (Retro UNIX 386 v1)
13148                                  <1>       ; 14/04/2013 - 08/03/2014 (Retro UNIX 8086 v1)
13149                                  <1>       ;
13150                                  <1>       ; 'swap' is routine that controls the swapping of processes
13151                                  <1>       ; in and out of core.
13152                                  <1>       ;
```

```
13153                                  <1>       ; TRDOS 386 (TRDOS v2.0) modification ->  ** 20/05/2016 **
13154                                  <1>       ;     * 3 different priority level is applied
13155                                  <1>       ;      (just as original unix v1)
13156                                  <1>       ;      1) high priority (event) run queue, 'runq_event'
13157                                  <1>       ;      2) normal priority (regular) run queue, 'runq_normal'
13158                                  <1>       ;      3) low priority (background) run queue, 'runq_backgroud'
13159                                  <1>       ;      'swap' code will run a process which has max. priority
13160                                  <1>         ;       (for earliest event at first)
13161                                  <1>       ;
13162                                  <1>       ; Retro UNIX 386 v1 modification ->
13163                                  <1>       ;       swap (software task switch) is performed by changing
13164                                  <1>       ;      user's page directory (u.pgdir) instead of segment change
13165                                  <1>       ;      as in Retro UNIX 8086 v1.
13166                                  <1>       ;
13167                                  <1>       ; RETRO UNIX 8086 v1 modification ->
13168                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
13169                                  <1>       ;      according to 8086 cpu (x86 real mode) architecture.
13170                                  <1>       ;      pdp-11 was using 64KB uniform memory while IBM PC
13171                                  <1>       ;      compatibles was using 1MB segmented memory
13172                                  <1>       ;      in 8086/8088 times.
13173                                  <1>       ;
13174                                  <1>       ; INPUTS ->
13175                                  <1>       ;    runq table - contains processes to run.
13176                                  <1>       ;    p.link - contains next process in line to be run.
13177                                  <1>       ;    u.uno - process number of process in core
13178                                  <1>       ;    s.stack - swap stack used as an internal stack for swapping.
13179                                  <1>       ; OUTPUTS ->
13180                                  <1>       ;    (original unix v1 -> present process to its disk block)
13181                                  <1>       ;    (original unix v1 -> new process into core ->
13182                                  <1>       ;        Retro Unix 8086 v1 -> segment registers changed
13183                                  <1>       ;        for new process)
13184                                  <1>       ;    u.quant = 3 (Time quantum for a process)
13185                                  <1>       ;      ((INT 1Ch count down speed -> 18.2 times per second)
13186                                  <1>       ;    RETRO UNIX 8086 v1 will use INT 1Ch (18.2 times per second)
13187                                  <1>       ;       for now, it will swap the process if there is not
13188                                  <1>       ;       a keyboard event (keystroke) (Int 15h, function 4Fh)
13189                                  <1>       ;       or will count down from 3 to 0 even if there is a
13190                                  <1>       ;         keyboard event locking due to repetitive key strokes.
13191                                  <1>       ;       u.quant will be reset to 3 for RETRO UNIX 8086 v1.
13192                                  <1>       ;
13193                                  <1>       ; ((Modified registers: EAX, EDX, EBX, ECX, ESI, EDI))
13194                                  <1>
13195                                  <1>       ;NOTE:
13196                                  <1>       ;High priority queue is the first for selecting a process to run.
13197                                  <1>       ;If there is not a process in high priority level run queue,
13198                                  <1>       ;a process in normal priority run queue will be selected
13199                                  <1>       ;or a proces in low priority run queue will be selected if normal
13200                                  <1>       ;priority level run queue is empty.
13201                                  <1>
13202                                  <1>       ; 21/05/2016 -(3 priority levels, 3 run queues)
13203 0001196D BE[52030300]            <1>       mov    esi, runq ; 'runq_event' ; high priority, 'run for event'
13204 00011972 C605[A8960100]03        <1>       mov    byte [priority], 3 ; high priority + 1
13205 00011979 31DB                    <1>       xor    ebx, ebx ; 02/01/2017
13206                                  <1> swap_0: ; 1: / search runq table for highest priority process
13207 0001197B 66AD                    <1>       lodsw  ; mov ax, [esi], add esi+2
13208                                  <1>       ;xor   ebx, ebx ; 02/05/2016
13209 0001197D 6621C0                  <1>       and    ax, ax ; are there any processes to run in this Q entry
13210 00011980 750E                    <1>       jnz    short swap_2
13211                                  <1>       ; 21/05/2026
13212                                  <1>       ; runq_normal = runq+2, runq_background = runq+4
13213 00011982 FE0D[A8960100]          <1>       dec    byte [priority] ; 3 -> 3, 2 -> 1, 1-> 0
13214 00011988 75F1                    <1>       jnz    short swap_0
13215                                  <1>       ;cmp   esi, runq+6  ; if zero compare address to end of table
13216                                  <1>       ;jb    short swap_0 ; if not at end, go back
13217                                  <1> swap_1:
13218                                  <1>       ; 02/05/2016
13219                                  <1>       ; 29/04/2016 (TRDOS 386 = TRDOS v2.0)
13220                                  <1>       ; No user process to run...
13221                                  <1>       ; Run the kernel process... MainProg: Internal Command Interpreter
13222 0001198A FEC0                    <1>       inc    al ; mov al, 1  ; process number of MainProg
13223 0001198C FEC3                    <1>       inc    bl ; mov bl, al ; 1
13224 0001198E EB1E                    <1>       jmp    short swap_4
13225                                  <1> swap_2:
13226                                  <1>       ; 21/05/2016
13227 00011990 FE0D[A8960100]          <1>       dec    byte [priority] ; priority level of present user/process
13228                                  <1>                            ; 0, 1, 2
13229 00011996 4E                      <1>       dec    esi
13230 00011997 4E                      <1>         dec     esi
13231                                  <1>       ;
13232 00011998 88C3                    <1>       mov    bl, al
13233 0001199A 38E0                    <1>       cmp    al, ah ; is there only 1 process in the queue to be run
13234 0001199C 740A                    <1>       je     short swap_3 ; yes
13235 0001199E 8AA3[9F000300]          <1>       mov    ah, [ebx+p.link-1]
13236 000119A4 8826                    <1>              mov    [esi], ah ; move next process in line into run queue
13237 000119A6 EB06                    <1>       jmp    short swap_4
13238                                  <1> swap_3:
13239 000119A8 6631D2                  <1>       xor    dx, dx
13240 000119AB 668916                  <1>       mov    [esi], dx ; zero the entry; no processes on the Q
13241                                  <1> swap_4:
13242 000119AE 8A25[B3030300]          <1>       mov    ah, [u.uno]
13243 000119B4 38C4                    <1>       cmp    ah, al ; is this process the same as the process in core?
13244 000119B6 743B                    <1>              je     short swap_8 ; yes, don't have to swap
13245 000119B8 08E4                    <1>       or     ah, ah  ; is the process # = 0
13246 000119BA 740D                    <1>              jz     short swap_6 ; 'sysexit'
13247                                  <1>       ;cmp   ah, al ;is this process the same as the process in core?
13248                                  <1>              ;je    short swap_8 ; yes, don't have to swap
13249 000119BC 8925[60030300]          <1>       mov    [u.usp], esp ; return  address for 'syswait' & 'sleep'
13250 000119C2 E834000000              <1>       call   wswap   ; write out core to disk
13251 000119C7 EB1C                    <1>       jmp    short swap_7
13252                                  <1> swap_6:
13253                                  <1>       ; Deallocate memory pages belong to the process
13254                                  <1>       ; which is being terminated.
13255                                  <1>       ; (Retro UNIX 386 v1 modification !)
13256                                  <1>       ;
13257 000119C9 53                      <1>       push   ebx
```

```
13258 000119CA A1[B8030300]       <1>       mov   eax, [u.pgdir]  ; page directory of the process
13259 000119CF 8B1D[BC030300]     <1>       mov   ebx, [u.ppgdir] ; page directory of the parent process
13260 000119D5 E83A43FFFF         <1>       call  deallocate_page_dir
13261 000119DA A1[B4030300]       <1>       mov   eax, [u.upage] ; 'user' structure page of the process
13262 000119DF E8D543FFFF         <1>       call  deallocate_page
13263 000119E4 5B                 <1>       pop   ebx
13264                             <1> swap_7:
13265 000119E5 C0E302             <1>       shl   bl, 2 ; * 4
13266 000119E8 8B83[BC000300]     <1>       mov   eax, [ebx+p.upage-4] ; the 'u' page of the new process
13267 000119EE E840000000         <1>       call  rswap ; read new process into core
13268                             <1> swap_8:
13269                             <1>       ; Retro UNIX  8086 v1 modification !
13270 000119F3 C605[A8030300]04   <1>       mov   byte [u.quant], time_count
13271 000119FA C3                 <1>       retn
13272                             <1>
13273                             <1> wswap:  ; < swap out, swap to disk >
13274                             <1>       ; 28/02/2017 (fnsave)
13275                             <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13276                             <1>       ; 09/05/2015 (Retro UNIX 386 v1)
13277                             <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
13278                             <1>       ; 'wswap' writes out the process that is in core onto its
13279                             <1>       ; appropriate disk area.
13280                             <1>       ;
13281                             <1>       ; Retro UNIX 386 v1 modification ->
13282                             <1>       ;       User (u) structure content and the user's register content
13283                             <1>       ;     will be copied to the process's/user's UPAGE (a page for
13284                             <1>       ;     saving 'u' structure and user registers for task switching).
13285                             <1>       ;     u.usp - points to kernel stack address which contains
13286                             <1>       ;             user's registers while entering system call.
13287                             <1>       ;     u.sp  - points to kernel stack address
13288                             <1>       ;             to return from system call -for IRET-.
13289                             <1>       ;     [u.usp]+32+16 = [u.sp]
13290                             <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
13291                             <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
13292                             <1>       ;
13293                             <1>       ; Retro UNIX 8086 v1 modification ->
13294                             <1>       ;       'swap to disk' is replaced with 'change running segment'
13295                             <1>       ;     according to 8086 cpu (x86 real mode) architecture.
13296                             <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
13297                             <1>       ;     compatibles was using 1MB segmented memory
13298                             <1>       ;     in 8086/8088 times.
13299                             <1>       ;
13300                             <1>       ; INPUTS ->
13301                             <1>       ;    u.break - points to end of program
13302                             <1>       ;    u.usp - stack pointer at the moment of swap
13303                             <1>       ;    core - beginning of process program
13304                             <1>       ;    ecore - end of core
13305                             <1>       ;    user - start of user parameter area
13306                             <1>       ;    u.uno - user process number
13307                             <1>       ;    p.dska - holds block number of process
13308                             <1>       ; OUTPUTS ->
13309                             <1>       ;    swp I/O queue
13310                             <1>       ;    p.break - negative word count of process
13311                             <1>       ;    r1 - process disk address
13312                             <1>       ;    r2 - negative word count
13313                             <1>       ;
13314                             <1>       ; RETRO UNIX 8086 v1 input/output:
13315                             <1>       ;
13316                             <1>       ; INPUTS ->
13317                             <1>       ;    u.uno - process number (to be swapped out)
13318                             <1>       ; OUTPUTS ->
13319                             <1>       ;    none
13320                             <1>       ;
13321                             <1>       ;   ((Modified registers: ECX, ESI, EDI))
13322                             <1>       ;
13323                             <1>
13324                             <1>       ; 28/02/2017
13325                             <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
13326                             <1>       ;jna   short wswp
13327 000119FB 803D[DA030300]00   <1>       cmp   byte [u.fpsave], 0 ; 28/02/2017
13328 00011A02 7606               <1>       jna   short wswp
13329 00011A04 DD35[DC030300]     <1>       fnsave [u.fpregs] ; save floating point registers (94 bytes)
13330                             <1> wswp:
13331 00011A0A 8B3D[B4030300]     <1>       mov   edi, [u.upage] ; process's user (u) structure page addr
13332 00011A10 B938000000         <1>       mov   ecx, (U_SIZE + 3) / 4
13333 00011A15 BE[5C030300]       <1>       mov   esi, user ; active user (u) structure
13334 00011A1A F3A5               <1>       rep   movsd
13335                             <1>       ;
13336 00011A1C 8B35[60030300]     <1>       mov   esi, [u.usp] ; esp (system stack pointer,
13337                             <1>                          ;      points to user registers)
13338 00011A22 8B0D[5C030300]     <1>       mov   ecx, [u.sp]  ; return address from the system call
13339                             <1>                          ; (for IRET)
13340                             <1>                          ; [u.sp] -> EIP (user)
13341                             <1>                          ; [u.sp+4]-> CS (user)
13342                             <1>                          ; [u.sp+8] -> EFLAGS (user)
13343                             <1>                          ; [u.sp+12] -> ESP (user)
13344                             <1>                          ; [u.sp+16] -> SS (user)
13345 00011A28 29F1               <1>       sub   ecx, esi     ; required space for user registers
13346 00011A2A 83C114             <1>       add   ecx, 20           ; +5 dwords to return from system call
13347                             <1>                          ; (for IRET)
13348 00011A2D C1E902             <1>       shr   ecx, 2
13349 00011A30 F3A5               <1>       rep   movsd
13350 00011A32 C3                 <1>       retn
13351                             <1>
13352                             <1> rswap:  ; < swap in, swap from disk >
13353                             <1>       ; 28/02/2017 (frstor)
13354                             <1>       ; 15/01/2017
13355                             <1>       ; 14/01/2017
13356                             <1>       ; 21/05/2016
13357                             <1>       ; 03/05/2016
13358                             <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13359                             <1>       ; 09/05/2015 - 15/09/2015 (Retro UNIX 386 v1)
13360                             <1>       ; 26/05/2013 - 08/03/2014 (Retro UNIX 8086 v1)
13361                             <1>       ; 'rswap' reads a process whose number is in r1,
13362                             <1>       ; from disk into core.
```

```
13363                                  <1>       ;
13364                                  <1>       ; Retro UNIX 386 v1 modification ->
13365                                  <1>       ;       User (u) structure content and the user's register content
13366                                  <1>       ;     will be restored from process's/user's UPAGE (a page for
13367                                  <1>       ;     saving 'u' structure and user registers for task switching).
13368                                  <1>       ;     u.usp - points to kernel stack address which contains
13369                                  <1>       ;             user's registers while entering system call.
13370                                  <1>       ;     u.sp  - points to kernel stack address
13371                                  <1>       ;             to return from system call -for IRET-.
13372                                  <1>       ;     [u.usp]+32+16 = [u.sp]
13373                                  <1>       ;     [u.usp] -> edi, esi, ebp, esp (= [u.usp]+32), ebx,
13374                                  <1>       ;             edx, ecx, eax, gs, fs, es, ds, -> [u.sp].
13375                                  <1>       ;
13376                                  <1>       ; RETRO UNIX 8086 v1 modification ->
13377                                  <1>       ;       'swap to disk' is replaced with 'change running segment'
13378                                  <1>       ;     according to 8086 cpu (x86 real mode) architecture.
13379                                  <1>       ;     pdp-11 was using 64KB uniform memory while IBM PC
13380                                  <1>       ;     compatibles was using 1MB segmented memory
13381                                  <1>       ;     in 8086/8088 times.
13382                                  <1>       ;
13383                                  <1>       ; INPUTS ->
13384                                  <1>       ;    r1 - process number of process to be read in
13385                                  <1>       ;    p.break - negative of word count of process
13386                                  <1>       ;    p.dska - disk address of the process
13387                                  <1>       ;    u.emt - determines handling of emt's
13388                                  <1>       ;    u.ilgins - determines handling of illegal instructions
13389                                  <1>       ; OUTPUTS ->
13390                                  <1>       ;    8 = (u.ilgins)
13391                                  <1>       ;    24 = (u.emt)
13392                                  <1>       ;    swp - bit 10 is set to indicate read
13393                                  <1>       ;             (bit 15=0 when reading is done)
13394                                  <1>       ;    swp+2 - disk block address
13395                                  <1>       ;    swp+4 - negative word count
13396                                  <1>       ;      ((swp+6 - address of user structure))
13397                                  <1>       ;
13398                                  <1>       ; RETRO UNIX 8086 v1 input/output:
13399                                  <1>       ;
13400                                  <1>       ; INPUTS ->
13401                                  <1>       ;    AL      - new process number (to be swapped in)
13402                                  <1>       ; OUTPUTS ->
13403                                  <1>       ;    none
13404                                  <1>       ;
13405                                  <1>       ;   ((Modified registers: EAX, ECX, ESI, EDI, ESP))
13406                                  <1>       ;
13407                                  <1>       ; Retro UNIX 386 v1 - modification ! 14/05/2015
13408 00011A33 89C6                   <1>       mov    esi, eax  ; process's user (u) structure page addr
13409 00011A35 B938000000             <1>       mov    ecx, (U_SIZE + 3) / 4
13410 00011A3A BF[5C030300]           <1>       mov    edi, user ; active user (u) structure
13411 00011A3F F3A5                   <1>       rep    movsd
13412 00011A41 58                     <1>       pop    eax    ; 'rswap' return address
13413                                  <1>       ;
13414                                  <1>       ;cli
13415 00011A42 8B3D[60030300]         <1>       mov    edi, [u.usp] ; esp (system stack pointer,
13416                                  <1>                           ;     points to user registers)
13417 00011A48 89FC                   <1>       mov    esp, edi     ; 14/01/2017
13418 00011A4A 8B0D[5C030300]         <1>       mov    ecx, [u.sp]  ; return address from the system call
13419                                  <1>                           ; (for IRET)
13420                                  <1>                           ; [u.sp] -> EIP (user)
13421                                  <1>                           ; [u.sp+4]-> CS (user)
13422                                  <1>                           ; [u.sp+8] -> EFLAGS (user)
13423                                  <1>                           ; [u.sp+12] -> ESP (user)
13424                                  <1>                           ; [u.sp+16] -> SS (user)
13425 00011A50 29F9                   <1>       sub    ecx, edi     ; required space for user registers
13426 00011A52 83C114                 <1>       add    ecx, 20           ; +5 dwords to return from system call
13427                                  <1>                           ; (for IRET)
13428 00011A55 C1E902                 <1>       shr    ecx, 2
13429 00011A58 F3A5                   <1>       rep    movsd
13430                                  <1>       ;mov   esp, [u.usp] ; 15/09/2015
13431                                  <1>       ;sti
13432                                  <1>       ; 28/02/2017
13433                                  <1>       ;cmp   byte [multi_tasking], 0  ; Musti tasking mode ?
13434                                  <1>       ;jna   short rswp_retn
13435 00011A5A 803D[DA030300]00       <1>       cmp    byte [u.fpsave], 0
13436 00011A61 7606                   <1>       jna    short rswp_retn
13437 00011A63 DD25[DC030300]         <1>       frstor [u.fpregs] ; restore floating point regs (94 bytes)
13438                                  <1> rswp_retn:
13439 00011A69 50                     <1>       push   eax    ; 'rswap' return address
13440 00011A6A C3                     <1>       retn
13441                                  <1>
13442                                  <1> putlu:
13443                                  <1>       ; 20/05/2016
13444                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13445                                  <1>       ; 10/05/2015 - 12/09/2015 (Retro UNIX 386 v1)
13446                                  <1>       ; 15/04/2013 - 23/02/2014 (Retro UNIX 8086 v1)
13447                                  <1>       ; 'putlu' is called with a process number in r1 and a pointer
13448                                  <1>       ; to lowest priority Q (runq+4) in r2. A link is created from
13449                                  <1>       ; the last process on the queue to process in r1 by putting
13450                                  <1>       ; the process number in r1 into the last process's link.
13451                                  <1>       ;
13452                                  <1>       ; INPUTS ->
13453                                  <1>       ;    r1 - user process number
13454                                  <1>       ;    r2 - points to lowest priority queue
13455                                  <1>       ;    p.dska - disk address of the process
13456                                  <1>       ;    u.emt - determines handling of emt's
13457                                  <1>       ;    u.ilgins - determines handling of illegal instructions
13458                                  <1>       ; OUTPUTS ->
13459                                  <1>       ;    r3 - process number of last process on the queue upon
13460                                  <1>       ;        entering putlu
13461                                  <1>       ;    p.link-1 + r3 - process number in r1
13462                                  <1>       ;    r2 - points to lowest priority queue
13463                                  <1>       ;
13464                                  <1>       ; ((Modified registers: EDX, EBX))
13465                                  <1>       ;
13466                                  <1>       ; / r1 = user process no.; r2 points to lowest priority queue
13467                                  <1>
```

```
13468                                  <1>       ; EBX = r2
13469                                  <1>       ; EAX = r1 (AL=r1b)
13470                                  <1>
13471                                  <1>       ; 20/05/2016
13472                                  <1>       ; AL = process number (1 to 16) // Retro UNIX 8086, 386 v1 //
13473                                  <1>       ;     (max. 16 processes available for current kernel version)
13474                                  <1>       ; EBX = run queue address ; 20/05/2016 (TRDOS 386)
13475                                  <1>             ; which is one of following addresses:
13476                                  <1>             ;  1) 'runq_event' high priority run queue
13477                                  <1>             ;  2) 'runq_normal' normal/regular priority run queue
13478                                  <1>             ;  3) 'runq_background' low priority run queue
13479                                  <1>
13480                                  <1>       ;mov   ebx, runq
13481 00011A6B 0FB613                 <1>       movzx       edx, byte [ebx]
13482 00011A6E 43                     <1>       inc    ebx
13483 00011A6F 20D2                   <1>       and    dl, dl
13484                                  <1>             ; tstb (r2)+ / is queue empty?
13485 00011A71 740A                   <1>             jz     short putlu_1
13486                                  <1>             ; beq 1f / yes, branch
13487 00011A73 8A13                   <1>       mov    dl, [ebx] ; 12/09/2015
13488                                  <1>             ; movb (r2),r3 / no, save the "last user" process number
13489                                  <1>                         ; / in r3
13490 00011A75 8882[9F000300]         <1>             mov    [edx+p.link-1], al
13491                                  <1>             ; movb r1,p.link-1(r3) / put pointer to user on
13492                                  <1>                         ; / "last users" link
13493 00011A7B EB03                   <1>       jmp    short putlu_2
13494                                  <1>             ; br 2f /
13495                                  <1> putlu_1: ; 1:
13496 00011A7D 8843FF                 <1>       mov    [ebx-1], al
13497                                  <1>                     ; movb r1,-1(r2) / user is only user;
13498                                  <1>                         ; / put process no. at beginning and at end
13499                                  <1> putlu_2: ; 2:
13500 00011A80 8803                   <1>       mov    [ebx], al
13501                                  <1>                     ; movb r1,(r2) / user process in r1 is now the last entry
13502                                  <1>                         ; / on the queue
13503 00011A82 88C2                   <1>       mov    dl, al
13504 00011A84 88B2[9F000300]         <1>         mov     [edx+p.link-1], dh ; 0
13505                                  <1>             ; dec r2 / restore r2
13506 00011A8A C3                     <1>         retn
13507                                  <1>             ; rts r0
13508                                  <1>
13509                                  <1> sysver:
13510                                  <1>       ; 29/04/2016 - TRDOS 386 (TRDOS v2.0)
13511 00011A8B C705[64030300]0002-     <1>       mov    dword [u.r0], 200h ; AH = major version, AL = minor version
13511 00011A93 0000                   <1>
13512 00011A95 E972BEFFFF             <1>       jmp    sysret
13513                                  <1>
13514                                  <1>
13515                                  <1> syspri: ; change running priority (of the process)
13516                                  <1>       ; 21/05/2016
13517                                  <1>       ; 20/05/2026 - TRDOS 386 (TRDOS v2.0)
13518                                  <1>       ; INPUT ->
13519                                  <1>       ;     BL = priority level
13520                                  <1>       ;        0 = low running priority (running on background)
13521                                  <1>       ;        1 = normal/regular priority (running as regular)
13522                                  <1>       ;        2 = high/event priority (running for event)
13523                                  <1>       ;        >2 = invalid, it will accepted as 2 (event)
13524                                  <1>       ;        0FFh = get/return current running priority only
13525                                  <1>       ; OUTPUT ->
13526                                  <1>       ;     * if current [u.pri] < 2
13527                                  <1>       ;       if BL input < 0FFh ->
13528                                  <1>       ;          [u.pri] is updated as in BL input (0,1,2)
13529                                  <1>       ;       if BL input = 0FFh -> AL = [u.pri] (current)
13530                                  <1>       ;
13531                                  <1>       ;     * if current [u.pri] = 2
13532                                  <1>       ;       if BL input < 0FFh -> cf = 1 & AL = 2
13533                                  <1>       ;       if BL input = 0FFh -> cf = 0 & AL = 2
13534                                  <1>       ;
13535                                  <1>       ;     NOTE:
13536                                  <1>       ;     If [u.pri] = 2, it can not be changed to 1 or 0;
13537                                  <1>       ;     because, run queue of the running process is unspecified
13538                                  <1>       ;     at this      stage. Process might be started by a timer event
13539                                  <1>       ;     or priority might be changed to high by previous
13540                                  <1>       ;     'syspri' system     call. In both cases, the process is in
13541                                  <1>       ;     'runq_normal' or 'runq_background' queue.
13542                                  <1>       ;     As result of this fact, when the [u.quant] time quantum
13543                                  <1>       ;     of the process is elapsed or 'sysrele' system call is
13544                                  <1>       ;     instructed by the process, 'tswap' ('tswitch') procedure
13545                                  <1>       ;     will be called (to 'swap' or 'switch' out the procedure)
13546                                  <1>       ;     and it will not call 'putlu' to add the (stopping)
13547                                  <1>       ;     process to relevant run queue when [u.pri] = 2.
13548                                  <1>       ;     (Otherwise, it would be possible to add process to
13549                                  <1>       ;     a run queue while it is already in a run queue, wrongly.)
13550                                  <1>       ;
13551                                  <1>       ;     If [u.pri]< 2, 'tswap/tswitch' procedure will call
13552                                  <1>       ;     'putlu' to add process to relevant run queue
13553                                  <1>       ;     according to [u.pri] value. ('runq_normal' for 1,
13554                                  <1>       ;     'runq_background' for 0).
13555                                  <1>       ;
13556                                  <1>       ;     If BL input >= 2 and < 0FFh while [u.pri] < 2,
13557                                  <1>       ;     process will be added to 'runq_normal' queue and
13558                                  <1>       ;     [u.pri] will be set to 2. (in 'syspri' system call)
13559                                  <1>       ;
13560                                  <1>
13561 00011A9A 29C0                   <1>       sub    eax, eax ; 0
13562 00011A9C A3[C8030300]           <1>       mov    [u.error], eax
13563                                  <1>
13564 00011AA1 A0[A9030300]           <1>       mov    al, [u.pri]
13565 00011AA6 A3[64030300]           <1>       mov    [u.r0], eax
13566                                  <1>
13567 00011AAB FEC3                   <1>       inc    bl
13568 00011AAD 0F8459BEFFFF           <1>       jz     sysret ; 0FFh -> 0, get priority level
13569                                  <1>
13570 00011AB3 3C02                   <1>       cmp    al, 2
13571 00011AB5 0F8331BEFFFF           <1>       jnb    error ; CF = 1 & AL = 2 (& last error = 0)
```

```
13572                                  <1>
13573 00011ABB FECB                    <1>       dec    bl
13574 00011ABD 80FB02                  <1>       cmp    bl, 2
13575 00011AC0 7602                    <1>       jna    short syspri_1
13576 00011AC2 B302                    <1>       mov    bl, 2
13577                                  <1> syspri_1:
13578 00011AC4 881D[A9030300]          <1>       mov    [u.pri], bl
13579 00011ACA 80FB02                  <1>       cmp    bl, 2
13580 00011ACD 0F8239BEFFFF            <1>         jb      sysret
13581                                  <1>
13582                                  <1>       ; here...
13583                                  <1>       ; Priority of current process has been changed to high
13584                                  <1>       ; ('run for event') but current process will be added to
13585                                  <1>       ; 'run as normal' queue. ('run for event' high priority
13586                                  <1>       ; queue is under control of timer -& RTC- interrupt only!)
13587                                  <1>       ;
13588                                  <1>       ; (Otherwise, process can fall into black hole!
13589                                  <1>       ; e.g. if it is not in waiting list and it has not got
13590                                  <1>       ; a timer event and it is not in a run queue!
13591                                  <1>       ; Because, when [u.pri] is 2, 'tswap/tswitch' will not
13592                                  <1>       ; add the stopping process to a run queue.)
13593                                  <1>
13594 00011AD3 A0[B3030300]            <1>       mov    al, [u.uno]
13595 00011AD8 BB[54030300]            <1>       mov    ebx, runq_normal ; normal priority !
13596                                  <1>                               ; [u.pri] is set to high
13597                                  <1>                               ; but 'runq_event' queue is set
13598                                  <1>                               ; only by the kernel's timer
13599                                  <1>                               ; event function (timer interrupt).
13600 00011ADD E889FFFFFF              <1>       call   putlu
13601 00011AE2 E925BEFFFF              <1>       jmp    sysret
13602                                  <1>
13603                                  <1> cpass: ; / get next character from user area of core and put it in AL (r1)
13604                                  <1>       ; 02/05/2016 - TRDOS 386 (TRDOS v2.0)
13605                                  <1>       ; 19/05/2015 - 18/10/2015 (Retro UNIX 386 v1)
13606                                  <1>       ; 14/08/2013 - 20/09/2013 (Retro UNIX 8086 v1)
13607                                  <1>       ; INPUTS ->
13608                                  <1>       ;     [u.base] = virtual address in user area
13609                                  <1>       ;     [u.count] = byte count (max.)
13610                                  <1>       ;     [u.pcount] = byte count in page (0 = reset)
13611                                  <1>       ; OUTPUTS ->
13612                                  <1>       ;     AL = the character which is pointed by [u.base]
13613                                  <1>       ;     zf = 1 -> transfer count has been completed
13614                                  <1>         ;
13615                                  <1>       ; ((Modified registers:  EAX, EDX, ECX))
13616                                  <1>       ;
13617 00011AE7 833D[88030300]00        <1>       cmp    dword [u.count], 0  ; have all the characters been transferred
13618                                  <1>                                  ; i.e., u.count, # of chars. left
13619 00011AEE 763F                    <1>       jna    short cpass_3    ; to be transferred = 0?) yes, branch
13620 00011AF0 FF0D[88030300]          <1>       dec    dword [u.count]         ; no, decrement u.count
13621                                  <1>         ; 19/05/2015
13622                                  <1>       ;(Retro UNIX 386 v1 - translation from user's virtual address
13623                                  <1>       ;                  to physical address
13624 00011AF6 66833D[C4030300]00      <1>       cmp    word [u.pcount], 0 ; byte count in page = 0 (initial value)
13625                                  <1>                               ; 1-4095 --> use previous physical base address
13626                                  <1>                               ; in [u.pbase]
13627 00011AFE 770E                    <1>       ja     short cpass_1
13628 00011B00 833D[BC030300]00        <1>       cmp     dword [u.ppgdir], 0 ; is the caller os kernel
13629 00011B07 7427                    <1>         je      short cpass_k       ; (sysexec, '/etc/init') ?  (MainProg)
13630 00011B09 E858FDFFFF              <1>       call   trans_addr_r
13631                                  <1> cpass_1:
13632 00011B0E 66FF0D[C4030300]        <1>       dec    word [u.pcount]
13633                                  <1> cpass_2:
13634 00011B15 8B15[C0030300]          <1>       mov    edx, [u.pbase]
13635 00011B1B 8A02                    <1>       mov    al, [edx]    ; take the character pointed to
13636                                  <1>                               ; by u.base and put it in r1
13637 00011B1D FF05[8C030300]          <1>       inc    dword [u.nread] ; increment no. of bytes transferred
13638 00011B23 FF05[84030300]          <1>       inc    dword [u.base]  ; increment the buffer address to point to the
13639                                  <1>                                 ; next byte
13640 00011B29 FF05[C0030300]          <1>       inc    dword [u.pbase]
13641                                  <1> cpass_3:
13642 00011B2F C3                      <1>       retn
13643                                  <1> cpass_k:
13644                                  <1>       ; 02/07/2015
13645                                  <1>       ; The caller is os kernel
13646                                  <1>       ; (get sysexec arguments from kernel's memory space)
13647 00011B30 8B1D[84030300]          <1>       mov    ebx, [u.base]
13648 00011B36 66C705[C4030300]00-      <1>         mov     word [u.pcount], PAGE_SIZE ; 4096
13648 00011B3E 10                      <1>
13649 00011B3F 891D[C0030300]          <1>       mov    [u.pbase], ebx
13650 00011B45 EBCE                    <1>       jmp    short cpass_2
13651                                  <1>
13652                                  <1> transfer_to_user_buffer: ; fast transfer
13653                                  <1>       ; 27/05/2016
13654                                  <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
13655                                  <1>       ;
13656                                  <1>       ; INPUT ->
13657                                  <1>       ;     ESI = source address in system space
13658                                  <1>       ;     EDI = user's buffer address
13659                                  <1>       ;     ECX = transfer (byte) count
13660                                  <1>       ;     [u.pgdir] = user's page directory
13661                                  <1>       ; OUTPUT ->
13662                                  <1>       ;     ECX = actual transfer count
13663                                  <1>       ;     cf = 1 -> error
13664                                  <1>       ;     [u.count] = remain byte count
13665                                  <1>       ;
13666                                  <1>       ; Modified registers: eax, ecx
13667                                  <1>       ;
13668                                  <1>
13669 00011B47 21C9                    <1>       and    ecx, ecx
13670 00011B49 743B                    <1>       jz     short ttub_4
13671                                  <1>
13672 00011B4B 890D[88030300]          <1>       mov    [u.count], ecx
13673                                  <1>
13674 00011B51 57                      <1>       push   edi
13675 00011B52 56                      <1>       push   esi
```

```
13676 00011B53 53                <1>       push   ebx
13677 00011B54 52                <1>       push   edx
13678 00011B55 51                <1>       push   ecx
13679                            <1>
13680 00011B56 89FB              <1>       mov    ebx, edi
13681 00011B58 81C300004000      <1>       add    ebx, CORE ; 27/05/2016
13682                            <1> ttub_1:
13683                            <1>       ; ebx = virtual (linear) address
13684                            <1>       ; [u.pgdir] = user's page directory
13685 00011B5E E89347FFFF        <1>             call   get_physical_addr_x ; get physical address
13686 00011B63 7222              <1>       jc     short ttub_5
13687                            <1>       ; eax = physical address
13688                            <1>       ; ecx = remain byte count in page (1-4096)
13689 00011B65 89C7              <1>       mov    edi, eax
13690 00011B67 A1[88030300]      <1>       mov    eax, [u.count]
13691 00011B6C 39C1              <1>       cmp    ecx, eax
13692 00011B6E 7602              <1>       jna    short ttub_2
13693 00011B70 89C1              <1>       mov    ecx, eax
13694                            <1> ttub_2:
13695 00011B72 29C8              <1>       sub    eax, ecx
13696 00011B74 01CB              <1>       add    ebx, ecx
13697 00011B76 F3A4              <1>       rep    movsb
13698 00011B78 A3[88030300]      <1>       mov    [u.count], eax
13699 00011B7D 09C0              <1>       or     eax, eax
13700 00011B7F 75DD              <1>       jnz    short ttub_1
13701                            <1> ttub_retn:
13702                            <1> tfub_retn:
13703 00011B81 59                <1>       pop    ecx ; transfer count = actual transfer count
13704                            <1> ttub_3:
13705 00011B82 5A                <1>       pop    edx
13706 00011B83 5B                <1>       pop    ebx
13707 00011B84 5E                <1>       pop    esi
13708 00011B85 5F                <1>       pop    edi
13709                            <1> ttub_4:
13710 00011B86 C3                <1>       retn
13711                            <1> ttub_5:
13712 00011B87 59                <1>       pop    ecx
13713 00011B88 2B0D[88030300]    <1>       sub    ecx, [u.count] ; actual transfer count
13714 00011B8E F9                <1>       stc
13715 00011B8F EBF1              <1>       jmp    short ttub_3
13716                            <1>
13717                            <1> transfer_from_user_buffer: ; fast transfer
13718                            <1>       ; 27/05/2016
13719                            <1>       ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
13720                            <1>       ;
13721                            <1>       ; INPUT ->
13722                            <1>       ;      ESI = user's buffer address
13723                            <1>       ;      EDI = destination address in system space
13724                            <1>       ;      ECX = transfer (byte) count
13725                            <1>       ;      [u.pgdir] = user's page directory
13726                            <1>       ; OUTPUT ->
13727                            <1>       ;      ecx = actual transfer count
13728                            <1>       ;      cf = 1 -> error
13729                            <1>       ;      [u.count] = remain byte count
13730                            <1>       ;
13731                            <1>       ; Modified registers: eax, ecx
13732                            <1>       ;
13733                            <1>
13734 00011B91 21C9              <1>       and    ecx, ecx
13735                            <1>       ;jz    short tfub_4
13736 00011B93 74F1              <1>       jz     short ttub_4
13737                            <1>
13738 00011B95 890D[88030300]    <1>       mov    [u.count], ecx
13739                            <1>
13740 00011B9B 57                <1>       push   edi
13741 00011B9C 56                <1>       push   esi
13742 00011B9D 53                <1>       push   ebx
13743 00011B9E 52                <1>       push   edx
13744 00011B9F 51                <1>       push   ecx
13745                            <1>
13746 00011BA0 89F3              <1>       mov    ebx, esi
13747 00011BA2 81C300004000      <1>       add    ebx, CORE ; 27/05/2016
13748                            <1> tfub_1:
13749                            <1>       ; ebx = virtual (linear) address
13750                            <1>       ; [u.pgdir] = user's page directory
13751 00011BA8 E84947FFFF        <1>             call   get_physical_addr_x ; get physical address
13752                            <1>       ;jc    short tfub_5
13753 00011BAD 72D8              <1>       jc     short ttub_5
13754                            <1>       ; eax = physical address
13755                            <1>       ; ecx = remain byte count in page (1-4096)
13756 00011BAF 89C6              <1>       mov    esi, eax
13757 00011BB1 A1[88030300]      <1>       mov    eax, [u.count]
13758 00011BB6 39C1              <1>       cmp    ecx, eax
13759 00011BB8 7602              <1>       jna    short tfub_2
13760 00011BBA 89C1              <1>       mov    ecx, eax
13761                            <1> tfub_2:
13762 00011BBC 29C8              <1>       sub    eax, ecx
13763 00011BBE 01CB              <1>       add    ebx, ecx
13764 00011BC0 F3A4              <1>       rep    movsb
13765 00011BC2 A3[88030300]      <1>       mov    [u.count], eax
13766 00011BC7 09C0              <1>       or     eax, eax
13767 00011BC9 75DD              <1>       jnz    short tfub_1
13768                            <1>
13769 00011BCB EBB4              <1>       jmp    short tfub_retn
13770                            <1>
13771                            <1> ;tfub_retn:
13772                            <1> ;     pop    ecx ; transfer count = actual transfer count
13773                            <1> ;tfub_3:
13774                            <1> ;     pop    edx
13775                            <1> ;     pop    ebx
13776                            <1> ;     pop    esi
13777                            <1> ;     pop    edi
13778                            <1> ;tfub_4:
13779                            <1> ;     retn
13780                            <1> ;tfub_5:
```

```
13781                                  <1> ;       pop     ecx
13782                                  <1> ;       sub     ecx, [u.count] ; actual transfer count
13783                                  <1> ;       stc
13784                                  <1> ;       jmp     short tfub_3
13785                                  <1>
13786                                  <1> sysfff: ; <Find First File>
13787                                  <1>         ; 17/10/2016
13788                                  <1>         ; 16/10/2016
13789                                  <1>         ; 15/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
13790                                  <1>         ;           -derived from TRDOS v1.0, INT_21H.ASM-
13791                                  <1>         ;            ("loc_INT21h_find_first_file")
13792                                  <1>         ; TRDOS 8086 (v1.0)
13793                                  <1>           ;    07/08/2011
13794                                  <1>           ;    Find First File
13795                                  <1>         ;       INPUT:
13796                                  <1>           ;         CX= Attributes
13797                                  <1>           ;         DS:DX= Pointer to filename
13798                                  <1>         ;       MSDOS OUTPUT:
13799                                  <1>         ;           DTA: (Default address: PSP offset 80h)
13800                                  <1>         ;           Offset  Descrription
13801                                  <1>         ;           0       Reserved for use find next file
13802                                  <1>         ;           21      Attribute of file found
13803                                  <1>         ;           22      Time stamp of file
13804                                  <1>         ;           24      Date stamp of file
13805                                  <1>         ;           26      File size in bytes
13806                                  <1>         ;           30      Filename and extension (zero terminated)
13807                                  <1>         ;       If cf = 1:
13808                                  <1>         ;           Error Codes: (in AX)
13809                                  <1>         ;              2 - File not found
13810                                  <1>         ;              18 - No more files
13811                                  <1>           ;
13812                                  <1>         ; TRDOS 386 (v2.0)
13813                                  <1>         ; 15/10/2016
13814                                  <1>         ;
13815                                  <1>           ; INPUT ->
13816                                  <1>           ;       CL = File attributes
13817                                  <1>         ;            bit 0 (1) - Read only file (R)
13818                                  <1>         ;             bit 1 (1) - Hidden file (H)
13819                                  <1>           ;             bit 2 (1) - System file (R)
13820                                  <1>         ;             bit 3 (1) - Volume label/name (V)
13821                                  <1>           ;             bit 4 (1) - Subdirectory (D)
13822                                  <1>         ;            bit 5 (1) - File has been archived (A)
13823                                  <1>         ;         CH = 0 -> Return basic parameters (24 bytes)
13824                                  <1>         ;         CH > 0 -> Return FindFile structure/table (128 bytes)
13825                                  <1>           ;          EBX = Pointer to filename (ASCIIZ) -path-
13826                                  <1>         ;         EDX = File parameters buffer address
13827                                  <1>         ;             (buffer size = 24 bytes if CH input = 0)
13828                                  <1>         ;             (buffer size = 128 bytes if CH input > 0)
13829                                  <1>         ;
13830                                  <1>         ; OUTPUT ->
13831                                  <1>         ;         EAX = 0 if CH input > 0
13832                                  <1>         ;         EAX = First cluster number of file if CH input = 0
13833                                  <1>         ;         EDX = File parameters table/structure address
13834                                  <1>         ;         Basic Parameters:
13835                                  <1>         ;             Offset  Description
13836                                  <1>         ;             ------ ---------------
13837                                  <1>         ;             0      File Attributes
13838                                  <1>         ;             1      Ambiguous filename chars are used sign
13839                                  <1>         ;                    (0 = filename fits exactly with request)
13840                                  <1>         ;                    (>0 = ambiguous filename chars are used)
13841                                  <1>         ;             2      Time stamp of file
13842                                  <1>         ;             4      Date stamp of file
13843                                  <1>         ;             6      File size in bytes
13844                                  <1>         ;             10     Short Filename (ASCIIZ, max. 13 bytes)
13845                                  <1>         ;             23     Longname Length (1-255) if existing
13846                                  <1>         ;
13847                                  <1>         ;           cf = 1 -> Error code in AL
13848                                  <1>         ;
13849                                  <1>         ; Modified Registers: EAX (at the return of system call)
13850                                  <1>         ;
13851                                  <1>         ; TR-DOS FindFile (FFF) Structure (128 bytes):
13852                                  <1>         ; 09/10/2011 (DIR.ASM) - 10/02/2016 (trdoskx.s)
13853                                  <1>         ;
13854                                  <1>         ; Offset     Parameter           Size
13855                                  <1>         ; ------     ------------------  --------
13856                                  <1>         ; 0          FindFile_Drv        1 byte
13857                                  <1>         ; 1          FindFile_Directory  65 bytes
13858                                  <1>         ; 66         FindFile_Name       13 bytes
13859                                  <1>         ; 79         FindFile_LongNameEntryLength 1 byte
13860                                  <1>         ;Above 80 bytes form
13861                                  <1>         ;TR-DOS Source/Destination File FullName Format/Structure
13862                                  <1>         ; 80         FindFile_AttributesMask 1 word
13863                                  <1>         ; 82         FindFile_DirEntry   32 bytes (*)
13864                                  <1>         ; 114        FindFile_DirFirstCluster 1 double word
13865                                  <1>         ; 118        FindFile_DirCluster 1 double word
13866                                  <1>         ; 122        FindFile_DirEntryNumber 1 word
13867                                  <1>         ; 124        FindFile_MatchCounter     1 word
13868                                  <1>         ; 126        FindFile_Reserved   1 word
13869                                  <1>         ; (*) MS-DOS, FAT 12-16-32 classic directory entry (32 bytes)
13870                                  <1>
13871                                  <1>         ;mov   [u.namep], ebx
13872                                  <1>         ; 16/10/2016
13873 00011BCD 8915[C8960100]        <1>         mov    [FFF_UBuffer], edx
13874 00011BD3 66890D[CD960100]      <1>         mov    [FFF_Attrib], cx ; [FFF_RType] = ch
13875                                  <1>                    ; Attributes in CL, return data type in CH
13876 00011BDA 89DE                  <1>         mov    esi, ebx
13877                                  <1>         ; file name is forced, change directory as temporary
13878                                  <1>         ;mov   ax, 1
13879                                  <1>         ;mov   [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
13880                                  <1>         ;call  set_working_path
13881 00011BDC E8E2130000            <1>         call   set_working_path_x ; 17/10/2016
13882 00011BE1 731D                  <1>         jnc    short sysfff_0
13883                                  <1>
13884 00011BE3 21C0                  <1>         and    eax, eax  ; 0 -> Bad Path!
13885 00011BE5 7505                  <1>         jnz    short sysfff_err
```

```
13886                                   <1>
13887                                   <1>       ; eax = 0
13888 00011BE7 B80C000000                <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
13889                                   <1> sysfff_err:
13890 00011BEC A3[64030300]              <1>       mov    [u.r0], eax
13891 00011BF1 A3[C8030300]              <1>       mov    [u.error], eax
13892 00011BF6 E89D140000                <1>         call       reset_working_path
13893 00011BFB E9ECBCFFFF                <1>       jmp    error
13894                                   <1>
13895                                   <1> sysfff_0:
13896                                   <1>       ;sub   ah, ah ; ah = 0
13897 00011C00 8A0424                    <1>       mov    al, [esp]
13898 00011C03 08C0                      <1>       or     al, al
13899 00011C05 7412                      <1>       jz     short sysfff_2
13900 00011C07 B410                      <1>       mov    ah, 10h
13901 00011C09 A808                      <1>       test   al, 08h
13902 00011C0B 7503                      <1>       jnz    short sysfff_1
13903 00011C0D 80CC08                    <1>       or     ah, 08h
13904                                   <1> sysfff_1:
13905 00011C10 2410                      <1>       and    al, 10h ; Directory
13906 00011C12 7405                      <1>       jz     short sysfff_2
13907 00011C14 80E408                    <1>       and    ah, 08h
13908 00011C17 30C0                      <1>       xor    al, al ; When a directory is searched,
13909                                   <1>                     ; filename will be returned even if
13910                                   <1>                     ; it is not a directory!
13911                                   <1>                     ; Because: (in order to prevent
13912                                   <1>                     ; creating a dir with existing file name)
13913                                   <1>                     ; Dir and file names must not be same!
13914                                   <1>                     ; (return attribute must be checked)
13915                                   <1> sysfff_2:
13916                                   <1>       ; AX = Attributes mask
13917                                   <1>             ; AL = AND mask (result must be equal to AL)
13918                                   <1>             ; AH = Negative AND mask (result must be ZERO)
13919                                   <1>       ; ESI = FindFile_Name address
13920                                   <1>
13921 00011C19 E87378FFFF                <1>       call   find_first_file
13922 00011C1E 72CC                      <1>       jc     short sysfff_err ; eax = 2 (File not found !)
13923                                   <1>
13924                                   <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
13925                                   <1>       ; EDI = Directory Buffer Directory Entry Location
13926                                   <1>       ; EAX = File Size
13927                                   <1>       ;  BL = Attributes of The File/Directory
13928                                   <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
13929                                   <1>       ;  DX > 0 : Ambiguous filename chars are used
13930                                   <1>
13931                                   <1> sysfff_3:
13932                                   <1>       ; 16/10/2016
13933 00011C20 668B0D[CD960100]          <1>       mov    cx, [FFF_Attrib]
13934                                   <1>       ; Attribs in CL, return data type in CH
13935                                   <1>
13936                                   <1>       ;or    cl, cl
13937                                   <1>       ;jz    short sysfff_4 ; 0 = No filter
13938 00011C27 80F1FF                    <1>       xor    cl, 0FFh
13939 00011C2A 20D9                      <1>       and    cl, bl
13940 00011C2C 7409                      <1>       jz     short sysfff_4
13941                                   <1>
13942                                   <1>       ;mov   eax, 2 ; 'file not found !' error
13943                                   <1>       ;jmp   short sysfff_err_1
13944                                   <1>
13945                                   <1>       ; 16/10/2016
13946 00011C2E E80D79FFFF                <1>       call   find_next_file
13947 00011C33 72B7                      <1>       jc     short sysfff_err ; eax = 12 (no more files !)
13948 00011C35 EBE9                      <1>       jmp    short sysfff_3
13949                                   <1>
13950                                   <1> sysfff_4:
13951 00011C37 20ED                      <1>       and    ch, ch ; [FFF_RType]
13952 00011C39 7412                      <1>       jz     short sysfff_5
13953 00011C3B B980000000                <1>       mov    ecx, 128 ; ; transfer length
13954 00011C40 880D[CC960100]            <1>       mov    [FFF_Valid], cl
13955                                   <1> sysfnf_11:
13956 00011C46 BE[7E930100]              <1>       mov    esi, FindFile_Drv
13957 00011C4B EB44                      <1>       jmp    short sysfff_6
13958                                   <1> sysfff_5:
13959                                   <1>       ;mov   esi, FindFile_DirEntry
13960 00011C4D B918000000                <1>       mov    ecx, 24  ; transfer length
13961 00011C52 880D[CC960100]            <1>       mov    [FFF_Valid], cl
13962                                   <1> sysfnf_12:
13963 00011C58 BF[889B0100]              <1>       mov    edi, DTA ; FFF data transfer address
13964                                   <1>       ;mov   al, [esi+DirEntry_Attr] ; 11
13965 00011C5D 88D8                      <1>       mov    al, bl ; File/Dir Attributes
13966 00011C5F 887F17                    <1>       mov    [edi+23], bh ; Longname length (0= none)
13967 00011C62 AA                        <1>       stosb
13968 00011C63 88D0                      <1>       mov    al, dl ; DL is for '?'
13969 00011C65 00F0                      <1>       add    al, dh ; DH is for '*'
13970                                   <1>       ; AL > 0 if ambiguous file name wildcards are used
13971 00011C67 AA                        <1>       stosb
13972 00011C68 8B4616                    <1>       mov    eax, [esi+DirEntry_WrtTime] ; 22
13973 00011C6B AB                        <1>         stosd      ; DirEntry_WrtTime & DirEntry_WrtDate
13974 00011C6C 8B461C                    <1>         mov  eax, [esi+DirEntry_FileSize] ; 28
13975 00011C6F AB                        <1>         stosd
13976 00011C70 668B4614                  <1>       mov    ax, [esi+DirEntry_FstClusHI] ; 20
13977 00011C74 66C1E010                  <1>       shl    ax, 16
13978 00011C78 668B461A                  <1>       mov    ax, [esi+DirEntry_FstClusLO] ; 26
13979 00011C7C A3[64030300]              <1>       mov    [u.r0], eax ; First Cluster
13980                                   <1>
13981                                   <1>         ;mov esi, FindFile_DirEntry
13982 00011C81 E84F140000                <1>       call   get_file_name
13983                                   <1>
13984 00011C86 8A0D[CC960100]            <1>       mov    cl, [FFF_Valid]
13985 00011C8C BE[889B0100]              <1>              mov    esi, DTA ; FFF data transfer address
13986                                   <1> sysfff_6:
13987 00011C91 8B3D[C8960100]            <1>       mov    edi, [FFF_UBuffer] ; user's buffer address (edx)
13988 00011C97 E8ABFEFFFF                <1>       call   transfer_to_user_buffer
13989                                   <1>
13990 00011C9C 890D[64030300]            <1>       mov    [u.r0], ecx ; actual transfer count
```

```
13991 00011CA2 E8F1130000          <1>         call       reset_working_path
13992 00011CA7 E960BCFFFF          <1>       jmp    sysret
13993                              <1>
13994                              <1> sysfnf: ; <Find Next File>
13995                              <1>       ; 16/10/2016 TRDOS 386 (TRDOS v2.0) feature only !
13996                              <1>       ;           -derived from TRDOS v1.0, INT_21H.ASM-
13997                              <1>       ;            ("loc_INT21h_find_next_file")
13998                              <1>       ; TRDOS 8086 (v1.0)
13999                              <1>         ;    07/08/2011
14000                              <1>         ;    Find First File
14001                              <1>       ;      INPUT:
14002                              <1>         ;        none
14003                              <1>       ;      MSDOS OUTPUT:
14004                              <1>       ;          DTA: (Default address: PSP offset 80h)
14005                              <1>       ;          Offset  Descrription
14006                              <1>       ;          0      Reserved for use find next file
14007                              <1>       ;          21     Attribute of file found
14008                              <1>       ;          22     Time stamp of file
14009                              <1>       ;          24     Date stamp of file
14010                              <1>       ;          26     File size in bytes
14011                              <1>       ;          30     Filename and extension (zero terminated)
14012                              <1>       ;      If cf = 1:
14013                              <1>       ;          Error Codes: (in AX)
14014                              <1>       ;             18 - No more files
14015                              <1>         ;
14016                              <1>       ; TRDOS 386 (v2.0)
14017                              <1>       ; 16/10/2016
14018                              <1>       ;
14019                              <1>         ; INPUT ->
14020                              <1>             ;         none
14021                              <1>       ; OUTPUT ->
14022                              <1>       ;         EAX = 0 if CH input of 'Find First File' > 0
14023                              <1>       ;         EAX = First cluster number of file
14024                              <1>       ;             if CH input of 'Find First File' = 0
14025                              <1>       ;        EDX = File parameters table/structure address
14026                              <1>       ;
14027                              <1>       ;          cf = 1 -> Error code in AL
14028                              <1>       ;
14029                              <1>       ; Modified Registers: EAX (at the return of system call)
14030                              <1>
14031                              <1>       ;
14032                              <1>       ; Note: If byte [FFF_Valid] = 0
14033                              <1>       ;      'sysfnf' will return with 'no more files' error.
14034                              <1>       ;      If byte [FFF_Valid] = 24
14035                              <1>       ;      'sysfnf' will return with 32 bytes basic parameters
14036                              <1>       ;      at the address which is in EDX.
14037                              <1>       ;      If byte [FFF_Valid] = 128
14038                              <1>       ;      'sysfnf' will return with 128 bytes Find File
14039                              <1>       ;      Structure/Table at the address which is in EDX.
14040                              <1>
14041 00011CAC 803D[CC960100]00    <1>       cmp    byte [FFF_Valid], 0
14042 00011CB3 7714                <1>       ja     short stsfnf_0
14043                              <1>       ; 'no more files !' error
14044 00011CB5 B80C000000          <1>       mov    eax, ERR_NO_MORE_FILES ; 12
14045 00011CBA A3[64030300]        <1>       mov    [u.r0], eax
14046 00011CBF A3[C8030300]        <1>       mov    [u.error], eax
14047 00011CC4 E923BCFFFF          <1>       jmp    error
14048                              <1> stsfnf_0:
14049                              <1>       ;cmp   byte [FFF_Valid], 128
14050                              <1>       ;je    short stsfnf_1
14051                              <1>       ;cmp   byte [FFF_Valid], 24
14052                              <1>       ;je    short stsfnf_1
14053                              <1>       ;mov   [FFF_Valid], 24 ; Default
14054                              <1> stsfnf_1:
14055 00011CC9 0FB61D[DE8A0100]    <1>       movzx  ebx, byte [Current_Drv]
14056 00011CD0 66891D[D2960100]    <1>       mov    [SWP_DRV], bx
14057 00011CD7 8A15[7E930100]      <1>       mov    dl, [FindFile_Drv]
14058 00011CDD 38DA                <1>       cmp    dl, bl
14059 00011CDF 750B                <1>       jne    short stsfnf_2
14060 00011CE1 86FB                <1>       xchg   bh, bl
14061 00011CE3 BE00010900          <1>       mov    esi, Logical_DOSDisks
14062 00011CE8 01DE                <1>       add    esi, ebx
14063 00011CEA EB0D                <1>       jmp    short sysfnf_3
14064                              <1>
14065                              <1> stsfnf_2:
14066 00011CEC FE05[D3960100]      <1>       inc    byte [SWP_DRV_chg]
14067                              <1>
14068 00011CF2 E8F763FFFF          <1>       call   change_current_drive
14069 00011CF7 7245                <1>       jc     short sysfnf_err_1 ; read error !
14070                              <1>                                 ; (do not stop, because
14071                              <1>                                 ; we don't have a
14072                              <1>                                 ; 'no more files'
14073                              <1>                                 ; -file not found- error,
14074                              <1>                                 ; next sysfnf system call
14075                              <1>                                 ; may solve the problem,
14076                              <1>                                 ; after re-placing the disk)
14077                              <1> sysfnf_3:
14078 00011CF9 A1[F4930100]        <1>       mov    eax, [FindFile_DirCluster]
14079 00011CFE 21C0                <1>       and    eax, eax
14080 00011D00 7550                <1>       jnz    short sysfnf_6
14081                              <1>
14082 00011D02 803D[DD8A0100]02    <1>       cmp    byte [Current_FATType], 2
14083 00011D09 772C                <1>       ja     short sysfnf_err_0 ; invalid, we neeed to stop !?
14084 00011D0B 803D[DD8A0100]01    <1>       cmp    byte [Current_FATType], 1
14085 00011D12 7223                <1>       jb     short sysfnf_err_0 ; invalid, we neeed to stop !?
14086                              <1>
14087 00011D14 3805[04920100]      <1>       cmp    byte [DirBuff_ValidData], al ; 0
14088 00011D1A 7608                <1>       jna    short sysfnf_4
14089                              <1>
14090 00011D1C 3B05[09920100]      <1>       cmp    eax, [DirBuff_Cluster] ; 0 ?
14091 00011D22 745E                <1>       je     short sysfnf_9
14092                              <1>
14093                              <1>       ;cmp   byte [Current_Dir_Level], 0
14094                              <1>         ;ja  short sysfnf_4
14095                              <1>         ;jna short sysfnf_9
```

```
14096                                  <1>
14097                                  <1> sysfnf_4:
14098 00011D24 FE05[D3960100]          <1>       inc   byte [SWP_DRV_chg]
14099 00011D2A E8A2B1FFFF              <1>       call  load_FAT_root_directory
14100 00011D2F 7351                    <1>       jnc   short sysfnf_9
14101                                  <1>       ; eax = error code (17, 'drv not ready or read error')
14102 00011D31 EB0B                    <1>       jmp   short sysfnf_err_1 ; read error ! (no FNF stop)
14103                                  <1>                                ; (if you want, try again,
14104                                  <1>                                ;  after re-placing the disk)
14105                                  <1> sysfnf_5:
14106 00011D33 3C0C                    <1>       cmp   al, 12 ; 'no more files' error
14107 00011D35 7507                    <1>       jne   short sysfnf_err_1 ; (no FNF stop -sysfnf will try
14108                                  <1>                                ;  to read the directory again,
14109                                  <1>                                ;  if the user calls sysfnf
14110                                  <1>                                ;  just after this error return-)
14111                                  <1>       ; (FNF stop -sysfnf will not try
14112                                  <1>       ;  to read the directory again-)
14113                                  <1>
14114                                  <1> sysfnf_err_0:
14115 00011D37 C605[CC960100]00        <1>       mov   byte [FFF_Valid], 0 ; FNF stop sign
14116                                  <1> sysfnf_err_1:
14117 00011D3E A3[64030300]            <1>       mov   [u.r0], eax
14118 00011D43 A3[C8030300]            <1>       mov   [u.error], eax
14119 00011D48 E84B130000              <1>       call  reset_working_path
14120 00011D4D E99ABBFFFF              <1>       jmp   error
14121                                  <1>
14122                                  <1> sysfnf_6:
14123 00011D52 803D[04920100]00        <1>       cmp   byte [DirBuff_ValidData], 0
14124 00011D59 7608                    <1>       jna   short sysfnf_7
14125                                  <1>
14126 00011D5B 3B05[09920100]          <1>       cmp   eax, [DirBuff_Cluster]
14127 00011D61 741F                    <1>       je    short sysfnf_9
14128                                  <1>
14129                                  <1> sysfnf_7:
14130 00011D63 FE05[D3960100]          <1>       inc   byte [SWP_DRV_chg]
14131 00011D69 803D[DD8A0100]01        <1>       cmp   byte [Current_FATType], 1
14132 00011D70 7309                    <1>       jnb   short sysfnf_8
14133                                  <1>
14134                                  <1>       ; Singlix (TRFS) File System
14135                                  <1>       ; (access via compatibility buffer)
14136 00011D72 E822B2FFFF              <1>       call  load_FS_sub_directory
14137 00011D77 7309                    <1>       jnc   short sysfnf_9
14138                                  <1>
14139 00011D79 EBC3                    <1>       jmp   short sysfnf_err_1 ; read error (no FNF stop)
14140                                  <1>
14141                                  <1> sysfnf_8:
14142 00011D7B E8DCB1FFFF              <1>       call  load_FAT_sub_directory
14143 00011D80 72BC                    <1>       jc    short sysfnf_err_1 ; read error (no FNF stop)
14144                                  <1>
14145                                  <1> sysfnf_9:
14146 00011D82 E8B977FFFF              <1>       call  find_next_file
14147 00011D87 72AA                    <1>       jc    short sysfnf_5
14148                                  <1>
14149 00011D89 A0[CD960100]            <1>       mov   al, [FFF_Attrib]
14150                                  <1>       ;or   al, al
14151                                  <1>       ;jz   short sysfnf_10 ; 0 = No filter
14152 00011D8E 34FF                    <1>       xor   al, 0FFh
14153 00011D90 20D8                    <1>       and   al, bl
14154 00011D92 75EE                    <1>       jnz   short sysfnf_9 ; search for next file until
14155                                  <1>                            ; an error return from
14156                                  <1>                            ; find_next_file procedure
14157                                  <1> sysfnf_10:
14158 00011D94 0FB60D[CC960100]        <1>         movzx     ecx, byte [FFF_Valid]
14159 00011D9B 80F980                  <1>       cmp   cl, 128 ; complete FindFile structure/table
14160 00011D9E 0F84A2FEFFFF            <1>       je    sysfnf_11
14161                                  <1>       ;cmp  cl, 24  ; basic parameters
14162                                  <1>       ;je   sysfnf_12
14163 00011DA4 E9AFFEFFFF              <1>       jmp   sysfnf_12
14164                                  <1>
14165                                  <1> writei:
14166                                  <1>       ; 26/10/2016
14167                                  <1>       ; 25/10/2016
14168                                  <1>       ; 23/10/2016
14169                                  <1>       ; 22/10/2016
14170                                  <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
14171                                  <1>       ; 19/05/2015 - 20/05/2015 (Retro UNIX 386 v1)
14172                                  <1>       ; 12/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
14173                                  <1>       ;
14174                                  <1>       ; Write data to file with first cluster number in EAX
14175                                  <1>       ;
14176                                  <1>       ; INPUTS ->
14177                                  <1>       ;    EAX - First cluster number of the file
14178                                  <1>       ;    EBX - File number (Open file index number)
14179                                  <1>       ;    u.count - byte count to be written
14180                                  <1>       ;    u.base - points to user buffer
14181                                  <1>       ;    u.fofp - points to dword with current file offset
14182                                  <1>       ;    i.size - file size
14183                                  <1>       ;    cdev - logical dos drive number of the file
14184                                  <1>       ; OUTPUTS ->
14185                                  <1>       ;    u.count - cleared
14186                                  <1>       ;    u.nread - accumulates total bytes passed back
14187                                  <1>       ;    i.size - new file size (if file byte offset overs file size)
14188                                  <1>       ;    u.fofp - points to u.off (with new offset value)
14189                                  <1>       ;
14190                                  <1>       ; (Retro UNIX Prototype : 11/11/2012 - 18/11/2012, UNIXCOPY.ASM)
14191                                  <1>       ; ((Modified registers: eax, edx, ebx, ecx, esi, edi, ebp))
14192                                  <1>
14193 00011DA9 31C9                    <1>       xor   ecx, ecx
14194 00011DAB 890D[8C030300]          <1>       mov   [u.nread], ecx  ; 0
14195 00011DB1 66890D[C4030300]        <1>       mov   [u.pcount], cx ; 19/05/2015
14196 00011DB8 390D[88030300]          <1>       cmp   [u.count], ecx
14197 00011DBE 7701                    <1>       ja    short writei_1
14198 00011DC0 C3                      <1>       retn
14199                                  <1> writei_1:
14200 00011DC1 881D[8C960100]          <1>       mov   [writei.ofn], bl ; Open file number
```

```
14201 00011DC7 880D[C7960100]      <1>       mov    [setfmod], cl ; 0 ; reset 'update lm date&time' sign
14202                              <1> dskw_0:
14203                              <1>       ; 26/10/2016
14204                              <1>       ; 22/10/2016, 23/10/2016, 25/10/2016
14205                              <1>       ; 19/10/2016 - TRDOS 386 (TRDOS v2.0)
14206                              <1>       ; 31/05/2015 - 25/07/2015 (Retro UNIX 386 v1)
14207                              <1>       ; 26/04/2013 - 20/09/2013 (Retro UNIX 8086 v1)
14208                              <1>       ;
14209                              <1>       ; 01/08/2013 (mkdir_w check)
14210 00011DCD E8D7000000          <1>       call   mget_w
14211                              <1>       ; eax = sector/block number
14212                              <1>
14213 00011DD2 8B1D[74030300]      <1>       mov     ebx, [u.fofp]
14214 00011DD8 8B13                <1>       mov    edx, [ebx]
14215 00011DDA 81E2FF010000        <1>       and    edx, 1FFh  ; / test the lower 9 bits of the file offset
14216 00011DE0 750C                <1>       jnz    short dskw_1 ; / if its non-zero, branch
14217                              <1>                           ; if zero, file offset = 0,
14218                              <1>                                  ; / 512, 1024,...(i.e., start of new block)
14219 00011DE2 813D[88030300]0002- <1>       cmp    dword [u.count], 512
14219 00011DEA 0000                <1>
14220                              <1>                                ; / if zero, is there enough data to fill
14221                              <1>                                ; / an entire block? (i.e., no. of
14222 00011DEC 7337                <1>       jnb    short dskw_2 ; / bytes to be written greater than 512.?
14223                              <1>                               ; / Yes, branch. Don't have to read block
14224                              <1> dskw_1: ; in as no past info. is to be saved
14225                              <1>       ; (the entire block will be overwritten).
14226                              <1>       ; 23/10/2016
14227                              <1>
14228 00011DEE BB[94070300]        <1>       mov    ebx, writei_buffer
14229                              <1>       ; esi = logical dos drive description table address
14230                              <1>       ; eax = sector number
14231                              <1>       ; ebx = buffer address (in kernel's memory space)
14232                              <1>       ; ecx = sector count
14233 00011DF3 B901000000          <1>       mov    ecx, 1
14234 00011DF8 E8A30D0000          <1>       call   disk_read
14235                              <1>       ;call  dskrd  ; / no, must retain old info..
14236                              <1>                               ; / Hence, read block 'r1' into an I/O buffer
14237 00011DFD 7326                <1>       jnc    short dskw_2
14238                              <1>
14239                              <1>       ; disk read error
14240 00011DFF B811000000          <1>       mov    eax, 17 ; drive not ready or READ ERROR !
14241                              <1> dskw_err: ; jump from disk write error
14242 00011E04 A3[64030300]        <1>       mov    [u.r0], eax
14243 00011E09 A3[C8030300]        <1>       mov    [u.error], eax
14244                              <1>
14245 00011E0E 803D[C7960100]00    <1>       cmp    byte [setfmod], 0
14246 00011E15 0F86D1BAFFFF        <1>       jna    error
14247                              <1>
14248 00011E1B E8AF030000          <1>       call   update_file_lmdt ; update last modif. date&time of the file
14249                              <1>       ;mov   byte [setfmod], 0
14250                              <1>
14251 00011E20 E9C7BAFFFF          <1>       jmp    error
14252                              <1>
14253                              <1> dskw_2: ; 3:
14254                              <1>       ; 23/10/2016
14255 00011E25 C605[68960100]01    <1>       mov    byte [writei.valid], 1 ; writei buffer contains valid data
14256 00011E2C 56                  <1>       push   esi ; logical dos drive description table address
14257                              <1>       ; EAX (r1) = block/sector number
14258                              <1>       ;call  wslot
14259                              <1>              ; jsr r0,wslot / set write and inhibit bits in I/O queue,
14260                              <1>                       ; / proc. status=0, r5 points to 1st word of data
14261 00011E2D 803D[C6030300]00    <1>       cmp    byte [u.kcall], 0
14262 00011E34 770F                <1>       ja     short dskw_4 ; zf=0 -> the caller is 'mkdir'
14263                              <1>       ;
14264 00011E36 66833D[C4030300]00  <1>       cmp    word [u.pcount], 0
14265 00011E3E 7705                <1>       ja     short dskw_4
14266                              <1> dskw_3:
14267                              <1>       ; [u.base] = virtual address to transfer (as source address)
14268 00011E40 E821FAFFFF          <1>       call   trans_addr_r ; translate virtual address to physical (r)
14269                              <1> dskw_4:
14270 00011E45 BB[94070300]        <1>       mov    ebx, writei_buffer
14271                              <1>       ; EBX (r5) = system (I/O) buffer address
14272 00011E4A E883FAFFFF          <1>       call   sioreg
14273                              <1>       ; ESI = file (user data) offset
14274                              <1>       ; EDI = sector (I/O) buffer offset
14275                              <1>       ; ECX = byte count
14276                              <1>       ;
14277 00011E4F F3A4                <1>       rep    movsb
14278                              <1>       ; 25/07/2015
14279                              <1>       ; eax = remain bytes in buffer
14280                              <1>         ;       (check if remain bytes in the buffer > [u.pcount])
14281 00011E51 09C0                <1>       or     eax, eax
14282 00011E53 75EB                <1>       jnz    short dskw_3 ; (page end before system buffer end!)
14283                              <1>
14284                              <1>       ; 23/10/2016
14285 00011E55 B101                <1>       mov    cl, 1
14286 00011E57 5E                  <1>       pop    esi
14287 00011E58 A1[6C960100]        <1>       mov    eax, [writei.sector]
14288                              <1>       ; esi = logical dos drive description table address
14289                              <1>       ; eax = sector number
14290                              <1>       ; ebx = writei buffer address
14291                              <1>       ; ecx = sector count
14292 00011E5D E82F0D0000          <1>       call   disk_write ; / yes, write the block
14293 00011E62 7307                <1>       jnc    short dskw_5
14294                              <1>
14295 00011E64 B812000000          <1>       mov    eax, 18 ; drive not ready or WRITE ERROR !
14296 00011E69 EB99                <1>       jmp    short dskw_err
14297                              <1>
14298                              <1> dskw_5:
14299                              <1>       ; 26/10/2016
14300 00011E6B 0FB61D[8C960100]    <1>       movzx  ebx, byte [writei.ofn] ; open file number
14301 00011E72 C0E302              <1>       shl    bl, 2 ; *4
14302 00011E75 8B83[5C9A0100]      <1>       mov    eax, [ebx+OF_POINTER]
14303 00011E7B 3B83[849A0100]      <1>       cmp    eax, [ebx+OF_SIZE]
14304 00011E81 7606                <1>       jna    short dskw_6
```

```
14305 00011E83 8983[849A0100]      <1>       mov    [ebx+OF_SIZE], eax
14306                              <1> dskw_6:
14307                              <1>       ;shr   bl, 2
14308 00011E89 833D[88030300]00    <1>         cmp     dword [u.count], 0 ; / any more data to write?
14309 00011E90 760A                <1>       jna    short dskw_7
14310 00011E92 A1[7C960100]        <1>       mov    eax, [writei.fclust]
14311 00011E97 E931FFFFFF          <1>       jmp    dskw_0 ; / yes, branch
14312                              <1> dskw_7:
14313                              <1>       ; update last modif. date&time of the file
14314                              <1>       ; (also updates file size as OF_SIZE)
14315 00011E9C E82E030000          <1>       call   update_file_lmdt
14316                              <1>       ;mov   byte [setfmod], 0
14317                              <1>
14318                              <1>       ; 03/08/2013
14319 00011EA1 C605[C6030300]00    <1>       mov    byte [u.kcall], 0
14320                              <1>       ; 23/10/2016
14321                              <1>       ;mov   eax, [writei.fclust]
14322 00011EA8 C3                  <1>       retn
14323                              <1>
14324                              <1> mget_w:
14325                              <1>       ; 02/11/2016
14326                              <1>       ; 01/11/2016
14327                              <1>       ; 23/10/2016, 31/10/2016
14328                              <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
14329                              <1>       ; 03/06/2015 (Retro UNIX 386 v1, 'mget', u.5s)
14330                              <1>       ; 22/03/2013 - 31/07/2013 (Retro UNIX 8086 v1)
14331                              <1>       ;
14332                              <1>       ; Get existing or (allocate) a new disk block for file
14333                              <1>       ;
14334                              <1>       ; INPUTS ->
14335                              <1>       ;    [u.fofp] = file offset pointer
14336                              <1>       ;    [i.size] = file size
14337                              <1>       ;    [u.count] = byte count
14338                              <1>       ;    EAX = First cluster
14339                              <1>       ;    [cdev] = Logical dos drive number
14340                              <1>       ;    [writei.ofn] = File Number
14341                              <1>       ;               (Open file index, 0 based)
14342                              <1>       ;    ([u.off] = file offset)
14343                              <1>       ; OUTPUTS ->
14344                              <1>       ;    EAX = logical sector number
14345                              <1>       ;    ESI = Logical Dos Drive Description Table address
14346                              <1>       ;
14347                              <1>       ; Modified registers: EDX, EBX, ECX, ESI, EDI, EBP
14348                              <1>
14349 00011EA9 8B35[74030300]      <1>         mov     esi, [u.fofp]
14350 00011EAF 8B2E                <1>       mov    ebp, [esi] ; u.off (or EBX*4+OF_POINTER)
14351                              <1>
14352 00011EB1 29C9                <1>       sub    ecx, ecx
14353 00011EB3 8A2D[46030300]      <1>       mov    ch, [cdev]
14354                              <1>
14355 00011EB9 BE00010900          <1>       mov    esi, Logical_DOSDisks
14356 00011EBE 01CE                <1>       add    esi, ecx
14357                              <1>
14358                              <1>       ; 31/10/2016
14359 00011EC0 89C3                <1>       mov    ebx, eax ; First Cluster or FDT address
14360                              <1>
14361 00011EC2 807E0300            <1>       cmp    byte [esi+LD_FATType], 0
14362 00011EC6 0F86DD010000        <1>       jna    mget_w_14 ; Singlix FS
14363                              <1>
14364 00011ECC 0FB74611            <1>       movzx  eax, word [esi+LD_BPB+BytesPerSec]
14365 00011ED0 0FB65613            <1>       movzx  edx, byte [esi+LD_BPB+SecPerClust]
14366 00011ED4 8815[6A960100]      <1>       mov    [writei.spc], dl  ; sectors per cluster
14367 00011EDA F7E2                <1>       mul    edx
14368                              <1>       ; edx = 0
14369                              <1>       ; eax = bytes per cluster (<= 65536)
14370                              <1>
14371                              <1>       ; 02/11/2016
14372 00011EDC 89C1                <1>       mov    ecx, eax
14373 00011EDE 48                  <1>       dec    eax
14374 00011EDF 66A3[70960100]      <1>       mov    [writei.bpc], ax
14375                              <1>
14376 00011EE5 89E8                <1>       mov    eax, ebp
14377 00011EE7 0305[88030300]      <1>       add    eax, [u.count] ; next file position
14378 00011EED 3B05[55040300]      <1>       cmp    eax, [i.size] ; <= file size ?
14379 00011EF3 0F86FC000000        <1>       jna    mget_w_4 ; no
14380                              <1>
14381 00011EF9 F7F1                <1>       div    ecx
14382 00011EFB A3[78960100]        <1>       mov    [writei.c_index], eax ; cluster index
14383                              <1>       ; edx = byte offset in cluster (<= 65535)
14384                              <1>       ;mov   [writei.offset], dx
14385                              <1>       ;shr   dx, 9 ; / 512
14386                              <1>       ;mov   [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14387                              <1>
14388 00011F00 29D2                <1>       sub    edx, edx ; 01/11/2016
14389 00011F02 8915[6C960100]      <1>       mov    [writei.sector], edx ; 0
14390 00011F08 668915[72960100]    <1>       mov    [writei.offset], dx  ; byte offset in cluster
14391 00011F0F 8815[6B960100]      <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14392                              <1>
14393 00011F15 89D8                <1>       mov    eax, ebx ; First Cluster
14394                              <1>
14395                              <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
14396 00011F17 3815[68960100]      <1>       cmp    byte [writei.valid], dl ; 0
14397 00011F1D 7624                <1>       jna    short mget_w_0
14398                              <1>
14399 00011F1F 8815[68960100]      <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
14400                              <1>
14401 00011F25 3B05[7C960100]      <1>       cmp    eax, [writei.fclust]
14402 00011F2B 7516                <1>       jne    short mget_w_0
14403                              <1>
14404 00011F2D 8A0D[46030300]      <1>       mov    cl, [cdev]
14405 00011F33 3A0D[69960100]      <1>       cmp    cl, [writei.drv]
14406 00011F39 7508                <1>       jne    short mget_w_0
14407                              <1>       ; [writei.l_clust] & [writei.l_index] are valid,
14408                              <1>       ;  we don't need to get last cluster & last cluster index
14409 00011F3B 8B0D[88960100]      <1>       mov    ecx, [writei.l_index]
```

```
14410 00011F41 EB64              <1>       jmp    short mget_w_2
14411                            <1> mget_w_0:
14412 00011F43 A3[7C960100]      <1>       mov    [writei.fclust], eax ; first cluster
14413                            <1>       ; edx = 0
14414 00011F48 A3[74960100]      <1>       mov    [writei.cluster], eax ; first cluster ; 01/11/2016
14415 00011F4D 8915[80960100]    <1>       mov    [writei.fs_index], edx ; 0 ; curret cluster index
14416                            <1>
14417                            <1>       ; FAT file system (FAT12, FAT16, FAT32)
14418 00011F53 E819B6FFFF        <1>       call   get_last_cluster
14419 00011F58 0F822B010000      <1>       jc     mget_w_err ; eax = error code
14420                            <1>
14421 00011F5E A3[84960100]      <1>       mov    [writei.lclust], eax ; last cluster
14422                            <1>
14423 00011F63 8B0D[A8940100]    <1>       mov    ecx, [glc_index] ; last cluster index
14424 00011F69 890D[88960100]    <1>       mov    [writei.l_index], ecx
14425                            <1>
14426 00011F6F A0[8C960100]      <1>       mov    al, [writei.ofn]
14427 00011F74 FEC0              <1>       inc    al
14428 00011F76 A2[C7960100]      <1>       mov    [setfmod], al ; update lm date&time sign
14429                            <1>
14430                            <1> mget_w_1:
14431 00011F7B 3B0D[78960100]    <1>       cmp    ecx, [writei.c_index]  ; last cluster index
14432 00011F81 7324              <1>       jnb    short mget_w_2 ; 01/11/2016
14433                            <1>
14434 00011F83 A1[84960100]      <1>       mov    eax, [writei.lclust]
14435                            <1>       ; EAX = Last cluster
14436 00011F88 E8F2B6FFFF        <1>       call   add_new_cluster
14437 00011F8D 0F82F6000000      <1>       jc     mget_w_err ; eax = error code
14438                            <1>       ; edx = 0
14439 00011F93 A3[84960100]      <1>       mov    [writei.lclust], eax ; (new) last cluster
14440 00011F98 8B0D[88960100]    <1>       mov    ecx, [writei.l_index]
14441 00011F9E 41                <1>       inc    ecx ; add 1 to last cluster index
14442 00011F9F 890D[88960100]    <1>       mov    [writei.l_index], ecx ; current last cluster index
14443                            <1>
14444 00011FA5 EBD4              <1>       jmp    short mget_w_1
14445                            <1>
14446                            <1> mget_w_2:
14447 00011FA7 89E9              <1>       mov    ecx, ebp
14448 00011FA9 030D[88030300]    <1>       add    ecx, [u.count]
14449 00011FAF 890D[55040300]    <1>       mov    [i.size], ecx ; save new file size
14450                            <1>       ;sub   edx, edx ; 0
14451                            <1>
14452 00011FB5 A0[46030300]      <1>       mov    al, [cdev]
14453 00011FBA A2[69960100]      <1>       mov    [writei.drv], al ; physical drive number
14454                            <1>       ; edx = 0
14455 00011FBF 89E8              <1>       mov    eax, ebp ; file offset
14456 00011FC1 0FB70D[70960100]  <1>       movzx  ecx, word [writei.bpc] ; bytes per cluster - 1
14457 00011FC8 41                <1>       inc    ecx ; bytes per cluster
14458 00011FC9 F7F1              <1>       div    ecx
14459                            <1>       ; edx = byte offset in cluster (<= 65535)
14460                            <1>       ; eax = cluster index
14461 00011FCB A3[78960100]      <1>       mov    [writei.c_index], eax
14462 00011FD0 668915[72960100]  <1>       mov    [writei.offset], dx
14463 00011FD7 66C1EA09          <1>       shr    dx, 9 ; / 512
14464 00011FDB 8815[6B960100]    <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14465                            <1>
14466                            <1> mget_w_3:
14467 00011FE1 3B05[88960100]    <1>       cmp    eax, [writei.l_index] ; last cluster index
14468 00011FE7 752A              <1>       jne    short mget_w_5
14469                            <1>
14470 00011FE9 A3[80960100]      <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
14471 00011FEE A1[84960100]      <1>       mov    eax, [writei.lclust] ; last cluster
14472 00011FF3 EB60              <1>       jmp    short mget_w_10
14473                            <1>
14474                            <1> mget_w_4: ; 02/11/2016
14475                            <1>       ; eax = next file position
14476 00011FF5 2B05[88030300]    <1>       sub    eax, [u.count] ; current file position
14477                            <1>       ; edx = 0
14478                            <1>       ; ecx = bytes per cluster
14479 00011FFB F7F1              <1>       div    ecx
14480 00011FFD A3[78960100]      <1>       mov    [writei.c_index], eax ; cluster index
14481 00012002 668915[72960100]  <1>       mov    [writei.offset], dx
14482 00012009 66C1EA09          <1>       shr    dx, 9 ; / 512
14483 0001200D 8815[6B960100]    <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14484                            <1>
14485                            <1> mget_w_5:
14486 00012013 21C0              <1>       and    eax, eax ; 0 = First Cluster's index number
14487 00012015 750C              <1>       jnz    short mget_w_6
14488                            <1>
14489 00012017 A3[80960100]      <1>       mov    [writei.fs_index], eax ; cluster index (for next check)
14490 0001201C A1[7C960100]      <1>       mov    eax, [writei.fclust] ; first cluster
14491 00012021 EB32              <1>       jmp    short mget_w_10
14492                            <1>
14493                            <1> mget_w_6:
14494 00012023 3B05[80960100]    <1>       cmp    eax, [writei.fs_index] ; current cluster index (>0)
14495 00012029 7507              <1>       jne    short mget_w_7
14496 0001202B A1[74960100]      <1>       mov    eax, [writei.cluster] ; current cluster
14497 00012030 EB3A              <1>       jmp    short mget_w_11
14498                            <1>
14499                            <1> mget_w_7:
14500 00012032 89C1              <1>       mov    ecx, eax
14501 00012034 2B0D[80960100]    <1>       sub    ecx, [writei.fs_index]
14502 0001203A 730D              <1>       jnc    short mget_w_8
14503                            <1>       ; get cluster by index from the first cluster
14504 0001203C A1[7C960100]      <1>       mov    eax, [writei.fclust]
14505 00012041 8B0D[78960100]    <1>       mov    ecx, [writei.c_index]
14506 00012047 EB05              <1>       jmp    short mget_w_9
14507                            <1>
14508                            <1> mget_w_8:
14509 00012049 A1[74960100]      <1>       mov    eax, [writei.cluster] ; beginning cluster
14510                            <1>       ; ecx = cluster sequence number after the beginning cluster
14511                            <1>       ; sub  edx, edx ; 0
14512                            <1>
14513                            <1> mget_w_9:
14514                            <1>       ; EAX = Beginning cluster
```

```
14515                                  <1>       ; EDX = Sector index in disk/file section
14516                                  <1>       ;       (Only for SINGLIX file system!)
14517                                  <1>       ; ECX = Cluster sequence number after the beginning cluster
14518                                  <1>       ; ESI = Logical DOS Drive Description Table address
14519 0001204E E832B7FFFF               <1>       call   get_cluster_by_index
14520 00012053 7234                     <1>       jc     short mget_w_err ; error code in EAX
14521                                  <1>       ; EAX = Cluster number
14522                                  <1> mget_w_10:
14523 00012055 A3[74960100]             <1>       mov    [writei.cluster], eax ; FDT number for Singlix File System
14524                                  <1>
14525 0001205A 807E0300                 <1>       cmp    byte [esi+LD_FATType], 0
14526 0001205E 7638                     <1>       jna    short mget_w_13
14527                                  <1>       ; 01/11/2016
14528 00012060 8B15[78960100]           <1>       mov    edx, [writei.c_index]
14529 00012066 8915[80960100]           <1>       mov    [writei.fs_index], edx
14530                                  <1> mget_w_11:
14531 0001206C 83E802                   <1>       sub    eax, 2
14532 0001206F 0FB615[6A960100]         <1>       movzx  edx, byte [writei.spc]
14533 00012076 F7E2                     <1>       mul    edx
14534                                  <1>
14535 00012078 034668                   <1>       add    eax, [esi+LD_DATABegin]
14536 0001207B 8A15[6B960100]           <1>       mov    dl, [writei.s_index]
14537 00012081 01D0                     <1>       add    eax, edx
14538                                  <1> mget_w_12:
14539 00012083 A3[6C960100]             <1>       mov    [writei.sector], eax
14540                                  <1>       ;; buffer validation must be done in writei
14541                                  <1>       ;;mov  byte [writei.valid], 1
14542 00012088 C3                       <1>       retn
14543                                  <1>
14544                                  <1> mget_w_err:
14545 00012089 A3[C8030300]             <1>       mov    [u.error], eax
14546 0001208E A3[64030300]             <1>       mov    [u.r0], eax
14547 00012093 E954B8FFFF               <1>       jmp    error
14548                                  <1>
14549                                  <1> mget_w_13:
14550                                  <1>       ; EAX = FDT number (Current Section)
14551                                  <1>       ; EDX = Sector index from the first section (0,1,2,3,4...)
14552 00012098 2B15[80960100]           <1>       sub    edx, [writei.fs_index]
14553                                  <1>       ; EDX = Sector index from current section
14554 0001209E 8915[80960100]           <1>       mov    [writei.fs_index], edx
14555 000120A4 40                       <1>       inc    eax ; the first data sector in FS disk section
14556 000120A5 01D0                     <1>       add    eax, edx
14557 000120A7 EBDA                     <1>       jmp    short mget_w_12
14558                                  <1>
14559                                  <1> mget_w_14:
14560 000120A9 8A4E12                   <1>       mov    cl, [esi+LD_FS_BytesPerSec+1]
14561 000120AC D0E9                     <1>       shr    cl, 1 ;  ; 1 for 512 bytes, 4 for 2048 bytes
14562 000120AE 880D[6A960100]           <1>       mov    [writei.spc], cl  ; sectors per cluster
14563                                  <1>       ; NOTE: writei bytes per sector value is always 512 !
14564 000120B4 66C705[70960100]00-      <1>       mov    word [writei.bpc], 512
14564 000120BC 02                       <1>
14565                                  <1>
14566 000120BD 89E9                     <1>       mov    ecx, ebp
14567 000120BF 030D[88030300]           <1>       add    ecx, [u.count] ; next file position
14568 000120C5 3B0D[55040300]           <1>       cmp    ecx, [i.size] ; <= file size ?
14569 000120CB 0F86C8000000             <1>       jna    mget_w_19 ; no
14570                                  <1>
14571 000120D1 29D2                     <1>       sub    edx, edx ; 0
14572 000120D3 8915[6C960100]           <1>       mov    [writei.sector], edx ; 0
14573 000120D9 668915[72960100]         <1>       mov    [writei.offset], dx  ; byte offset in cluster
14574 000120E0 8815[6B960100]           <1>       mov    [writei.s_index], dl ; sector index in cluster (0 to spc -1)
14575                                  <1>
14576 000120E6 C1E909                   <1>       shr    ecx, 9 ; 1 cluster = 512 bytes
14577 000120E9 890D[78960100]           <1>       mov    [writei.c_index], ecx ; section/cluster index
14578                                  <1>
14579 000120EF 89D8                     <1>       mov    eax, ebx ; FDT number (First FDT address)
14580                                  <1>
14581                                  <1>       ; is this the 1st mget_w or a next mget_w call ? (by 'writei')
14582 000120F1 3815[68960100]           <1>       cmp    byte [writei.valid], dl ; 0
14583 000120F7 7624                     <1>       jna    short mget_w_15
14584                                  <1>
14585 000120F9 8815[68960100]           <1>       mov    byte [writei.valid], dl ; 0 ; reset ('writei' will set it)
14586                                  <1>
14587 000120FF 3B05[7C960100]           <1>       cmp    eax, [writei.fclust]
14588 00012105 7516                     <1>       jne    short mget_w_15
14589                                  <1>
14590 00012107 8A0D[46030300]           <1>       mov    cl, [cdev]
14591 0001210D 3A0D[69960100]           <1>       cmp    cl, [writei.drv]
14592 00012113 7508                     <1>       jne    short mget_w_15
14593                                  <1>       ; [writei.l_clust] & [writei.l_index] are valid,
14594                                  <1>       ;  we don't need to get last cluster & last cluster index
14595 00012115 8B0D[88960100]           <1>       mov    ecx, [writei.l_index]
14596 0001211B EB49                     <1>       jmp    short mget_w_17
14597                                  <1> mget_w_15:
14598 0001211D A3[7C960100]             <1>       mov    [writei.fclust], eax ; first section (FDT number)
14599                                  <1>       ; edx = 0
14600 00012122 8915[74960100]           <1>       mov    [writei.cluster], edx ; 0 ; current section
14601 00012128 8915[80960100]           <1>       mov    [writei.fs_index], edx ; 0 ; curret section index
14602                                  <1>
14603                                  <1>       ; eax = FDT number (section 0 header address)
14604 0001212E E87CB6FFFF               <1>       call   get_last_section
14605 00012133 0F8250FFFFFF             <1>       jc     mget_w_err ; eax = error code
14606                                  <1>
14607 00012139 8915[80960100]           <1>       mov    [writei.fs_index], edx ; sector index in last section
14608                                  <1>
14609 0001213F A3[84960100]             <1>       mov    [writei.lclust], eax ; last section address
14610                                  <1>
14611 00012144 8B0D[A8940100]           <1>       mov    ecx, [glc_index] ; last section index
14612 0001214A 890D[88960100]           <1>       mov    [writei.l_index], ecx
14613                                  <1>
14614 00012150 A0[8C960100]             <1>       mov    al, [writei.ofn]
14615 00012155 FEC0                     <1>       inc    al
14616 00012157 A2[C7960100]             <1>       mov    [setfmod], al ; update lm date&time sign
14617                                  <1>
14618                                  <1> mget_w_16:
```

```
14619                                   <1>       ; edx = (existing) last section (sector) index
14620 0001215C 8B0D[78960100]           <1>       mov    ecx, [writei.c_index] ; final section (sector) index
14621 00012162 29D1                     <1>       sub    ecx, edx
14622 00012164 7633                     <1>       jna    short mget_w_19
14623                                   <1>        ; ecx = sector count
14624                                   <1> mget_w_17:
14625 00012166 A1[84960100]             <1>       mov    eax, [writei.lclust]
14626                                   <1>       ; ESI = Logical dos drv desc. table address
14627                                   <1>         ; EAX = Last section
14628                                   <1>         ; (ECX = 0 for directory)
14629                                   <1>         ; ECX = sector count (except FDT)
14630 0001216B E802ACFFFF               <1>       call   add_new_fs_section
14631 00012170 7312                     <1>       jnc    short mget_w_18
14632                                   <1>
14633                                   <1>       ; If error number = 27h (insufficient disk space)
14634                                   <1>       ; it is needed to check free consequent sectors
14635                                   <1>       ; (1 data sector at least and +1 section header sector)
14636                                   <1>
14637 00012172 83F827                   <1>       cmp    eax, 27h
14638 00012175 0F850EFFFFFF             <1>       jne    mget_w_err ; eax = error code
14639                                   <1>
14640                                   <1>       ; ecx = count of free consequent sectors
14641                                   <1>       ; ecx must be > 1 (1 data + 1 header sector)
14642 0001217B 49                       <1>       dec    ecx
14643 0001217C 0F8407FFFFFF             <1>       jz     mget_w_err
14644 00012182 EBE2                     <1>       jmp    short mget_w_17
14645                                   <1>
14646                                   <1> mget_w_18:
14647 00012184 A3[84960100]             <1>       mov    [writei.lclust], eax ; (new) last section
14648                                   <1>       ; ecx = sector count (except section header)
14649 00012189 8B15[88960100]           <1>       mov    edx, [writei.l_index]
14650 0001218F 01CA                     <1>       add    edx, ecx ; add sector count to index
14651 00012191 8915[88960100]           <1>       mov    [writei.l_index], edx
14652 00012197 EBC3                     <1>       jmp    short mget_w_16
14653                                   <1>
14654                                   <1> mget_w_19:
14655 00012199 89E9                     <1>       mov    ecx, ebp
14656 0001219B 030D[88030300]           <1>       add    ecx, [u.count]
14657 000121A1 890D[55040300]           <1>       mov    [i.size], ecx ; save new file size
14658                                   <1>       ;sub   edx, edx ; 0
14659                                   <1>
14660 000121A7 A0[46030300]             <1>       mov    al, [cdev]
14661 000121AC A2[69960100]             <1>       mov    [writei.drv], al ; physical drive number
14662                                   <1>       ; edx = 0
14663 000121B1 89E8                     <1>       mov    eax, ebp ; file offset
14664 000121B3 89C2                     <1>       mov    edx, eax
14665                                   <1>       ; 1 cluster = 512 bytes (for Singlix FS)
14666 000121B5 C1E809                   <1>       shr    eax, 9  ; / 512
14667 000121B8 81E2FF010000             <1>       and    edx, 1FFh
14668                                   <1>       ; edx = byte offset in cluster/sector (<= 511)
14669                                   <1>       ; eax = section (sector/cluster) index
14670 000121BE A3[78960100]             <1>       mov    [writei.c_index], eax
14671 000121C3 668915[72960100]         <1>       mov    [writei.offset], dx
14672                                   <1>       ;mov   byte [writei.s_index], 0 ; sector index in cluster
14673 000121CA E912FEFFFF               <1>       jmp    mget_w_3
14674                                   <1>
14675                                   <1> update_file_lmdt: ; & update file size
14676                                   <1>       ; 26/10/2016
14677                                   <1>       ; 24/10/2016
14678                                   <1>       ; 23/10/2016
14679                                   <1>       ; 22/10/2016 - TRDOS 386 (TRDOS v2.0)
14680                                   <1>       ;
14681                                   <1>       ; Update last modification date&time of file
14682                                   <1>       ; (call from syswrite -> writei)
14683                                   <1>       ; ((also updates file size)) // 26/10/2016
14684                                   <1>       ;
14685                                   <1>       ; INPUT:
14686                                   <1>       ;      byte [setfmod] = open file number
14687                                   <1>       ; OUTPUT:
14688                                   <1>       ;      cf = 0 -> success !
14689                                   <1>       ;      cf = 1 -> lmdt update has been failed!
14690                                   <1>       ;
14691                                   <1>       ; Modified registers: eax, ebx, ecx, edx, esi, edi
14692                                   <1>       ;
14693                                   <1>
14694                                   <1>       ;cmp   byte [setfmod], 0
14695                                   <1>       ;jna   short uflmdt_2 ; nothing to do
14696                                   <1>
14697 000121CF 31C0                     <1>       xor    eax, eax
14698                                   <1>
14699 000121D1 0FB61D[C7960100]         <1>       movzx  ebx, byte [setfmod]
14700 000121D8 FECB                     <1>       dec    bl ; open file index number (0 based)
14701                                   <1>
14702 000121DA 8AA3[349A0100]           <1>       mov    ah, [ebx+OF_DRIVE]
14703 000121E0 BE00010900               <1>       mov    esi, Logical_DOSDisks
14704 000121E5 01C6                     <1>       add    esi, eax
14705 000121E7 C0E302                   <1>       shl    bl, 2 ; *4
14706 000121EA 8B8B[0C9A0100]           <1>       mov    ecx, [ebx+OF_FCLUSTER] ; first cluster
14707 000121F0 8B93[D49A0100]           <1>       mov    edx, [ebx+OF_DIRCLUSTER] ; dir cluster
14708                                   <1>
14709 000121F6 D0EB                     <1>       shr    bl, 1 ; /2
14710 000121F8 0FB7BB[749B0100]         <1>       movzx  edi, word [ebx+OF_DIRENTRY]
14711                                   <1>
14712 000121FF 803D[04920100]01         <1>       cmp    byte [DirBuff_ValidData], 1
14713 00012206 726E                     <1>       jb     short uflmdt_4
14714                                   <1>
14715 00012208 A0[02920100]             <1>       mov    al, [DirBuff_DRV]
14716 0001220D 2C41                     <1>       sub    al, 'A'
14717 0001220F 38E0                     <1>       cmp    al, ah
14718 00012211 7563                     <1>       jne    short uflmdt_4 ; different drive
14719 00012213 8A4603                   <1>       mov    al, [esi+LD_FATType]
14720 00012216 3A05[03920100]           <1>       cmp    al, [DirBuff_FATType]
14721 0001221C 755B                     <1>       jne    short uflmdt_5 ; different FS type
14722 0001221E 3B15[09920100]           <1>       cmp    edx, [DirBuff_Cluster]
14723 00012224 7553                     <1>       jne    short uflmdt_5 ; different cluster
```

```
14724                                  <1>
14725                                  <1> uflmdt_1:
14726                                  <1>       ; Directory buffer is ready here!
14727                                  <1>       ; OF_FCLUSTER must be compared/verified
14728 00012226 BE00000800              <1>       mov   esi, Directory_Buffer
14729 0001222B 66C1E705                <1>       shl   di, 5 ; dir entry index * 32
14730 0001222F 01FE                    <1>       add   esi, edi ; offset
14731                                  <1>       ;
14732 00012231 F6460B18                <1>       test  byte [esi+DirEntry_Attr], 18h ; Vol & Dir
14733 00012235 750F                    <1>       jnz   short uflmdt_2 ; not a valid file !
14734 00012237 668B4614                <1>       mov   ax, [esi+DirEntry_FstClusHI]
14735 0001223B C1E010                  <1>       shl   eax, 16
14736 0001223E 668B461A                <1>       mov   ax, [esi+DirEntry_FstClusLO]
14737 00012242 39C8                    <1>       cmp   eax, ecx ; same first cluster ?
14738 00012244 7407                    <1>       je    short uflmdt_3 ; yes, it is OK !!!
14739                                  <1>
14740                                  <1> uflmdt_2:
14741                                  <1>       ; save directory buffer if has modified/changed sign
14742                                  <1>       ; (It is good to save dir buff even if the searched
14743                                  <1>       ; directory entry is not found !?)
14744 00012246 E87B98FFFF              <1>       call  save_directory_buffer
14745 0001224B F9                      <1>       stc   ; update failed
14746 0001224C C3                      <1>       retn
14747                                  <1>
14748                                  <1> uflmdt_3:
14749                                  <1>       ; Update directory entry
14750                                  <1>       ; 26/10/2016
14751 0001224D D0E3                    <1>       shl   bl, 1 ; *2
14752 0001224F 8B83[849A0100]          <1>       mov   eax, [ebx+OF_SIZE] ; file size
14753 00012255 89461C                  <1>       mov   [esi+DirEntry_FileSize], eax
14754                                  <1>       ;
14755 00012258 E8CB97FFFF              <1>       call  convert_current_date_time
14756                                  <1>       ; OUTPUT -> DX = Date in dos dir entry format
14757                                  <1>       ;       AX = Time in dos dir entry format
14758 0001225D 66894616                <1>       mov   [esi+DirEntry_WrtTime], ax
14759 00012261 66895618                <1>       mov   [esi+DirEntry_WrtDate], dx
14760 00012265 66895612                <1>       mov   [esi+DirEntry_LastAccDate], dx
14761 00012269 C605[04920100]02        <1>       mov   byte [DirBuff_ValidData], 2
14762 00012270 E85198FFFF              <1>       call  save_directory_buffer
14763 00012275 C3                      <1>       retn
14764                                  <1>
14765                                  <1> uflmdt_4:
14766                                  <1>       ; Directory buffer sector read&write
14767                                  <1>       ; 23/10/2016
14768                                  <1>       ;
14769 00012276 8A4603                  <1>       mov   al, [esi+LD_FATType]
14770                                  <1> uflmdt_5:
14771 00012279 BB[9C090300]            <1>       mov   ebx, rw_buffer ; Common r/w sector buffer addr
14772                                  <1>
14773 0001227E 20C0                    <1>       and   al, al ; 0 = Singlix FS
14774 00012280 0F8492000000            <1>       jz    uflmdt_11
14775                                  <1>
14776 00012286 21D2                    <1>       and   edx, edx
14777 00012288 7521                    <1>       jnz   short uflmdt_9
14778                                  <1>
14779 0001228A 3C02                    <1>       cmp   al, 2  ; 3 = FAT32
14780 0001228C 771A                    <1>       ja    short uflmdt_8
14781                                  <1>
14782 0001228E 89F8                    <1>       mov   eax, edi ; directory entry index number
14783 00012290 66C1E804                <1>       shr   ax, 4 ; 16 entries per sector
14784 00012294 034664                  <1>       add   eax, [esi+LD_ROOTBegin]
14785                                  <1>       ; eax = root directory sector
14786                                  <1> uflmdt_6:
14787 00012297 50                      <1>       push  eax ; * ; disk sector address
14788 00012298 51                      <1>       push  ecx ; first cluster
14789 00012299 B901000000              <1>       mov   ecx, 1
14790                                  <1>       ; ecx = sector count
14791 0001229E E8FD080000              <1>       call  disk_read
14792 000122A3 59                      <1>       pop   ecx
14793 000122A4 731A                    <1>       jnc   short uflmdt_10
14794 000122A6 58                      <1>       pop   eax ; *
14795                                  <1> uflmdt_7:
14796 000122A7 C3                      <1>       retn
14797                                  <1>
14798                                  <1> uflmdt_8:
14799 000122A8 8B5632                  <1>       mov   edx, [esi+LD_BPB+FAT32_RootFClust]
14800                                  <1> uflmdt_9:
14801 000122AB 83FA02                  <1>       cmp   edx, 2
14802 000122AE 72F7                    <1>       jb    short uflmdt_7 ; invalid, nothing to do
14803                                  <1>
14804 000122B0 83EA02                  <1>       sub   edx, 2
14805 000122B3 89D0                    <1>       mov   eax, edx
14806 000122B5 0FB65613                <1>       movzx edx, byte [esi+LD_BPB+SecPerClust]
14807 000122B9 F7E2                    <1>       mul   edx
14808 000122BB 034668                  <1>       add   eax, [esi+LD_DATABegin]
14809                                  <1>       ; eax = sub directory (data) sector
14810 000122BE EBD7                    <1>       jmp   short uflmdt_6
14811                                  <1>
14812                                  <1> uflmdt_10:
14813                                  <1>       ; Directory sector buffer is ready here!
14814                                  <1>       ; OF_FCLUSTER must be compared/verified
14815                                  <1>       ; edi = dir entry index number (<= 2047)
14816 000122C0 6683E70F                <1>       and   di, 0Fh ; 16 entries per sector
14817 000122C4 66C1E705                <1>       shl   di, 5 ; dir entry index * 32
14818 000122C8 81C7[9C090300]          <1>       add   edi, rw_buffer
14819                                  <1>       ;
14820 000122CE F6470B18                <1>       test  byte [edi+DirEntry_Attr], 18h ; Vol & Dir
14821 000122D2 0F856EFFFFFF            <1>       jnz   uflmdt_2 ; not a valid file !
14822 000122D8 668B5714                <1>       mov   dx, [edi+DirEntry_FstClusHI]
14823 000122DC C1E210                  <1>       shl   edx, 16
14824 000122DF 668B571A                <1>       mov   dx, [edi+DirEntry_FstClusLO]
14825 000122E3 39CA                    <1>       cmp   edx, ecx ; same first cluster ?
14826 000122E5 0F855BFFFFFF            <1>       jne   uflmdt_2 ; no !?
14827                                  <1>
14828                                  <1>       ; Update directory entry
```

```
14829 000122EB E83897FFFF          <1>       call   convert_current_date_time
14830                              <1>       ; OUTPUT -> DX = Date in dos dir entry format
14831                              <1>         ;       AX = Time in dos dir entry format
14832 000122F0 66894716            <1>       mov    [edi+DirEntry_WrtTime], ax
14833 000122F4 66895718            <1>       mov    [edi+DirEntry_WrtDate], dx
14834 000122F8 66895712            <1>       mov    [edi+DirEntry_LastAccDate], dx
14835                              <1>
14836 000122FC 58                  <1>       pop    eax ; *
14837                              <1>
14838 000122FD BB[9C090300]        <1>       mov    ebx, rw_buffer ; Common r/w sector buffer addr
14839 00012302 B901000000          <1>       mov    ecx, 1
14840                              <1>       ; esi = logical dos description table address
14841                              <1>       ; eax = disk sector number/address (LBA)
14842                              <1>       ; ecx = sector count
14843                              <1>       ; ebx = buffer address
14844 00012307 E885080000          <1>       call   disk_write
14845 0001230C 0F8234FFFFFF        <1>       jc     uflmdt_2
14846                              <1>
14847                              <1>       ; save directory buffer if has modified/changed sign
14848 00012312 E8AF97FFFF          <1>       call   save_directory_buffer
14849 00012317 C3                  <1>       retn
14850                              <1>
14851                              <1> uflmdt_11:
14852                              <1>       ; 24/10/2016
14853                              <1>       ; Update last modification date & time of a file
14854                              <1>       ; on a disk with Singlix File System.
14855                              <1>       ;
14856                              <1>       ; (Method: Read the FDT -File Description Table-
14857                              <1>       ; sector of the file and update the lmdt data fields,
14858                              <1>       ; then write FDT sector to the disk.
14859                              <1>       ; /// It is easy but there is compatibility buffer
14860                              <1>       ; method also for changing directory entry data and
14861                              <1>       ; also there are some programming issues for Singlix
14862                              <1>       ; file system (TRFS), which are not completed yet!)
14863                              <1>       ;
14864                              <1>       ; Not ready yet ! (24/10/2016)
14865                              <1>       ; /// Temporary code for error return ! ///
14866 00012318 31C0                <1>       xor    eax, eax
14867 0001231A F9                  <1>       stc
14868 0001231B C3                  <1>       retn
14869                              <1>
14870                              <1> sysalloc:
14871                              <1>       ; 14/10/2017
14872                              <1>       ; 20/08/2017, 01/09/2017
14873                              <1>       ; 20/02/2017, 04/03/2017, 15/05/2017
14874                              <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
14875                              <1>       ; (TRDOS 386 feature only!)
14876                              <1>       ;
14877                              <1>       ; Allocate Contiguous Memory Block/Pages (for user)
14878                              <1>       ; (System call for DMA Buffer allocation etc.)
14879                              <1>       ;
14880                              <1>       ; INPUT ->
14881                              <1>       ;     EBX = Virtual address (for user)
14882                              <1>       ;          (Physical memory block/aperture
14883                              <1>       ;          will be mapped to this virtual address)
14884                              <1>       ;     ECX = Byte Count
14885                              <1>       ;          (will be rounded up to page border)
14886                              <1>       ;     If ECX = 0
14887                              <1>       ;         System call will return with an error (cf=1)
14888                              <1>       ;         but ECX will contain maximum size of
14889                              <1>       ;         available memory aperture and physical
14890                              <1>       ;         (beginning) address of that aperture
14891                              <1>       ;         (which have maximum size) will be in EAX.
14892                              <1>       ;     EDX = Upper limit of the requested physical memory
14893                              <1>       ;           block/pages.
14894                              <1>       ;           (The last byte address of the memory aperture
14895                              <1>       ;            must not be equal to or above this limit.)
14896                              <1>       ;     If EDX = 0
14897                              <1>       ;        there is NOLIMIT !
14898                              <1>       ;     If EDX = 0FFFFFFFFh (-1)
14899                              <1>       ;        ESI = Lower Limit !
14900                              <1>       ;            (Beginning of the block must not be 'less'
14901                              <1>       ;            than this.) (Must be equal to or above...)
14902                              <1>       ;        EDI = Upper Limit !
14903                              <1>       ;            (End of the block must be !less! than this)
14904                              <1>       ;            (The last byte addr of the memory aperture
14905                              <1>       ;            must not be equal to or above this limit.)
14906                              <1>       ;
14907                              <1>       ; OUTPUT ->
14908                              <1>       ;     If CF = 0
14909                              <1>       ;     EAX = Physical address of the allocated memory block
14910                              <1>       ;     ECX = Allocated bytes (as rounded up to page borders)
14911                              <1>       ;     EBX = Virtual address (as rounded up)
14912                              <1>       ;     IF CF = 1
14913                              <1>       ;         Requested (size of) Memory block could not be
14914                              <1>       ;         allocated to the user!
14915                              <1>       ;     IF CF = 1 & EAX = 0 (Insufficient memory error!)
14916                              <1>       ;        ECX = Total number of free bytes
14917                              <1>       ;              (not size of available contiguous bytes!)
14918                              <1>       ;     If CF = 1 & EAX > 0
14919                              <1>       ;        there is not a memory aperture with requested size
14920                              <1>       ;        but total free mem is not less than requested size.
14921                              <1>       ;        EAX = Physical addr of available memory aperture
14922                              <1>       ;             with max size
14923                              <1>       ;              (but it doesn't fit to the conditions!)
14924                              <1>       ;        ECX = Size of available memory aperture in bytes.
14925                              <1>       ;     If CF = 1 -> EAX = 0FFFFFFFFh
14926                              <1>       ;        Conditions/Parameters are wrong !
14927                              <1>       ;        ECX is same with input value.
14928                              <1>       ;
14929                              <1>       ; Note:      Previously allocated pages will be deallocated if
14930                              <1>       ;      new allocation conditions are met.
14931                              <1>       ;
14932                              <1>       ; Note: u.break control may be included in future versions
14933                              <1>       ;
```

```
14934                                  <1>
14935 0001231C 31C0                    <1>       xor    eax, eax ; 0
14936                                  <1>       ; 14/10/2017
14937 0001231E 4A                      <1>       dec    edx ; is there a limit ?
14938 0001231F 7810                    <1>       js     short sysalloc_1 ;  0 -> 0FFFFFFFFh -> NO LIMIT
14939 00012321 42                      <1>       inc    edx ; > 0
14940                                  <1>       ; Check upper address limit
14941                                  <1>       ;(round up to page borders)
14942 00012322 81C1FF0F0000            <1>       add    ecx, PAGE_SIZE-1 ; 4095
14943 00012328 6681E100F0              <1>       and    cx, ~PAGE_OFF ; not 4095
14944 0001232D 39CA                    <1>       cmp    edx, ecx ; upper limit - block size
14945 0001232F 7224                    <1>       jb     short sysalloc_err
14946                                  <1> sysalloc_1:
14947                                  <1>       ; EAX = Beginning address (physical)
14948                                  <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
14949                                  <1>       ; ECX = Number of bytes to be allocated
14950 00012331 E84F41FFFF              <1>       call   allocate_memory_block
14951 00012336 721D                    <1>       jc     short sysalloc_err
14952                                  <1>       ; 01/09/2017
14953 00012338 29C2                    <1>       sub    edx, eax ; upper limit address - beginning address
14954 0001233A 760F                    <1>       jna    short sysalloc_3 ; begin addr not less than the limit
14955 0001233C 39CA                    <1>       cmp    edx, ecx
14956 0001233E 720B                    <1>       jb     short sysalloc_3 ; end address overs the limit
14957                                  <1> sysalloc_2:
14958                                  <1>       ; EAX = Beginning (physical) addr of the allocated mem block
14959                                  <1>       ; ECX = Num of allocated bytes (rounded up to page borders)
14960 00012340 50                      <1>       push   eax ; * ; 04/03/2017
14961                                  <1>       ; Here, requested contiquous memory pages have been allocated
14962                                  <1>       ; on Memory Allocation Table but user's page directory
14963                                  <1>       ; and page tables have not been updated yet!
14964 00012341 51                      <1>       push   ecx ; **
14965                                  <1>       ; ebx = virtual address (will be rounded up to page border)
14966                                  <1>       ; ecx = number of bytes to be deallocated
14967                                  <1>       ;      will be adjusted to ebx+ecx round down - ebx round up
14968 00012342 E89944FFFF              <1>       call   deallocate_user_pages
14969 00012347 731F                    <1>       jnc    short sysalloc_4 ; EAX = Deallocated memory bytes
14970 00012349 59                      <1>       pop    ecx ; **
14971 0001234A 58                      <1>       pop    eax ; *
14972                                  <1> sysalloc_3:
14973                                  <1>       ; error !
14974                                  <1>       ; restore Memory Allocation Table Content
14975 0001234B E84243FFFF              <1>       call   deallocate_memory_block
14976 00012350 31C0                    <1>       xor    eax, eax ; 0
14977 00012352 48                      <1>       dec    eax ; 0FFFFFFFFh ; 15/05/2017
14978 00012353 EB09                    <1>       jmp    short sysalloc_wrong
14979                                  <1> sysalloc_err:
14980 00012355 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
14981 0001235B 894D18                  <1>       mov    [ebp+24], ecx ; return to user with ecx value
14982                                  <1> sysalloc_wrong:
14983                                  <1>       ; eax = 0FFFFFFFFh
14984 0001235E A3[64030300]            <1>       mov    [u.r0], eax
14985 00012363 E984B5FFFF              <1>       jmp    error
14986                                  <1> sysalloc_4:
14987 00012368 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
14988 0001236E 894518                  <1>       mov    [ebp+24], eax ; return to user with ecx value
14989 00012371 895D10                  <1>       mov    [ebp+16], ebx ; new value of ebx (rounded up)
14990 00012374 89C1                    <1>       mov    ecx, eax ; byte count (from 'deallocate_user_pages')
14991 00012376 5A                      <1>       pop    edx ; ** ; discard (another) byte count
14992 00012377 58                      <1>       pop    eax ; *
14993 00012378 A3[64030300]            <1>       mov    [u.r0], eax ; physical address
14994                                  <1>
14995 0001237D 51                      <1>       push   ecx ; 20/08/2017
14996                                  <1>       ;
14997                                  <1>       ; Write newly allocated contiguous (physical) pages
14998                                  <1>       ; on page dir and page tables of current user/process
14999                                  <1>       ; as PRESENT, USER, WRITABLE
15000                                  <1>       ; (then clear allocated pages)
15001 0001237E E85245FFFF              <1>       call   allocate_user_pages
15002                                  <1>       ;jnc   sysret ; OK! return to process with success...
15003                                  <1>
15004                                  <1>       ; 20/08/2017 ('sysdma' modification)
15005 00012383 59                      <1>       pop    ecx
15006 00012384 A1[64030300]            <1>       mov    eax, [u.r0]   ; physical address (of the block)
15007                                  <1>
15008 00012389 721D                    <1>       jc     short sysalloc_6
15009                                  <1>
15010 0001238B 833D[DCA00100]FF        <1>       cmp    dword [dma_addr], 0FFFFFFFFh ; -1
15011 00012392 0F8274B5FFFF            <1>       jb     sysret
15012                                  <1>
15013 00012398 A3[DCA00100]            <1>       mov    [dma_addr], eax ; save dma address for sysdma
15014 0001239D 890D[E0A00100]          <1>       mov    [dma_size], ecx ; save dma buff size for sysdma
15015                                  <1>
15016 000123A3 E964B5FFFF              <1>       jmp    sysret
15017                                  <1>
15018                                  <1> sysalloc_6:
15019                                  <1>       ;
15020                                  <1>       ; unexpected error ! insufficient memory !? conflict !?
15021                                  <1>       ; (!!?there is not a free page for a new page table?!!)
15022                                  <1>       ; We need to terminate process with error message !!!
15023                                  <1>       ;
15024 000123A8 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
15025 000123AE 8B4D18                  <1>       mov    ecx, [ebp+24] ; byte count
15026                                  <1>
15027                                  <1>       ; 20/08/2017
15028                                  <1>       ;mov   eax, [u.r0]   ; physical address (of the block)
15029                                  <1>
15030                                  <1>       ;
15031                                  <1>       ; restore Memory Allocation Table Content
15032 000123B1 E8DC42FFFF              <1>       call   deallocate_memory_block
15033                                  <1>       ;
15034 000123B6 803D[DA6F0000]03        <1>       cmp    byte [CRT_MODE], 3 ; 80x25 text mode?
15035 000123BD 7407                    <1>       je     short sysalloc_7 ; yes
15036                                  <1>       ; Current mode is VGA (or CGA graphics) mode,
15037                                  <1>       ; We need to return to text mode for displaying
15038                                  <1>       ; error message just before 'sysexit'.
```

```
15039 000123BF B003                <1>       mov   al, 3
15040 000123C1 E8AAF7FEFF          <1>       call  _set_mode
15041                              <1> sysalloc_7:
15042 000123C6 BE[07440100]        <1>       mov   esi, beep_Insufficient_Memory ; error message
15043 000123CB E89551FFFF          <1>       call  print_msg ; print/display the message
15044 000123D0 B801000000          <1>       mov   eax, 1 ; ax=1 is needed for 'sysexit' procedure
15045 000123D5 E9B9B6FFFF          <1>       jmp   sysexit ; and terminate the process !
15046                              <1>
15047                              <1> sysdalloc:
15048                              <1>       ; 19/02/2017 - TRDOS 386 (TRDOS v2.0)
15049                              <1>       ; (TRDOS 386 feature only!)
15050                              <1>       ;
15051                              <1>       ; Deallocate Memory Block/Pages (for user)
15052                              <1>       ; (Complementary call for sysalloc.)
15053                              <1>       ;
15054                              <1>       ; INPUT ->
15055                              <1>       ;     EBX = Virtual address (for user)
15056                              <1>       ;           (will be rounded up to page border)
15057                              <1>       ;     ECX = Byte Count
15058                              <1>       ;          (will be adjusted to page borders)
15059                              <1>       ;     If ICX = 0
15060                              <1>       ;        nothing to do
15061                              <1>       ;     If EBX + ECX > User's ESP
15062                              <1>       ;        nothing to do
15063                              <1>       ;
15064                              <1>       ; Note: u.break control may be included in future versions
15065                              <1>       ;
15066                              <1>       ; OUTPUT ->
15067                              <1>       ;     If CF = 0
15068                              <1>       ;        EAX = Deallocated memory bytes
15069                              <1>       ;        EBX = Virtual address (as rounded up)
15070                              <1>       ;     IF CF = 1
15071                              <1>       ;        EAX = 0
15072                              <1>       ;
15073                              <1>       ; Note:      Main purpose of this call is to deallocate/release
15074                              <1>       ;     previously allocated (physically) contiguous memory
15075                              <1>       ;     pages but beginning (virtual) address may not be
15076                              <1>       ;     followed by physically contiguous pages. So, this
15077                              <1>       ;     system call will deallocate user's virtually
15078                              <1>       ;     contiguous memory pages. Also, there is not any
15079                              <1>       ;     objections to use this system call without sysalloc
15080                              <1>       ;     system call; only possible objection is to lost data
15081                              <1>       ;     within user's memory space, if the beginning address
15082                              <1>       ;     and size is not proper.
15083                              <1>       ;
15084                              <1>       ; Note: Empty page tables will not be deallocated!!!
15085                              <1>       ;       (they will be deallocated at process termination)
15086                              <1>       ;
15087                              <1>       ; Note: When the program terminates itself or when it is
15088                              <1>       ;     terminated by operating system kernel, all allocated
15089                              <1>       ;     memory pages will be deallocated during termination
15090                              <1>       ;     stage. So, 'sysdalloc' is not necessary except
15091                              <1>       ;     forgiving memory block to other programs/processes.
15092                              <1>       ;
15093 000123DA 8B15[5C030300]      <1>       mov   edx, [u.sp]
15094 000123E0 8B420C              <1>       mov   eax, [edx+12] ; user's stack pointer
15095 000123E3 29C8                <1>       sub   eax, ecx ; esp - byte count
15096 000123E5 24FC                <1>       and   al, 0FCh ; dword alignment
15097 000123E7 39D8                <1>       cmp   eax, ebx
15098 000123E9 7220                <1>       jb    short sysdalloc_err ; deallocation overlaps with stack
15099                              <1>
15100 000123EB 31C0                <1>       xor   eax, eax
15101 000123ED 21C9                <1>       and   ecx, ecx
15102 000123EF 7407                <1>       jz    short sysdalloc_2
15103                              <1>
15104 000123F1 E8EA43FFFF          <1>       call  deallocate_user_pages
15105 000123F6 7213                <1>       jc    short sysdalloc_err
15106                              <1>
15107                              <1> sysdalloc_2:
15108 000123F8 A3[64030300]        <1>       mov   [u.r0], eax
15109 000123FD 8B2D[60030300]      <1>       mov   ebp, [u.usp]
15110 00012403 895D10              <1>       mov   [ebp+16], ebx ; new value of ebx
15111 00012406 E901B5FFFF          <1>       jmp   sysret
15112                              <1>
15113                              <1> sysdalloc_err:
15114 0001240B A3[64030300]        <1>       mov   [u.r0], eax ; 0
15115 00012410 E9D7B4FFFF          <1>       jmp   error
15116                              <1>
15117                              <1> syscalbac:
15118                              <1>       ; SYS CALLBACK
15119                              <1>       ; 03/08/2020
15120                              <1>       ; 16/04/2017
15121                              <1>       ; 14/04/2017
15122                              <1>       ; 13/04/2017
15123                              <1>       ; 28/02/2017
15124                              <1>       ; 26/02/2017
15125                              <1>       ; 24/02/2017
15126                              <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
15127                              <1>       ; (TRDOS 386 feature only!)
15128                              <1>       ;
15129                              <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
15130                              <1>       ;
15131                              <1>       ; INPUT ->
15132                              <1>       ;     BL = IRQ number (Hardware interrupt request number)
15133                              <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
15134                              <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
15135                              <1>       ;          (numbers >15 are invalid)
15136                              <1>       ;
15137                              <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
15138                              <1>       ;          1 = Link IRQ by using Signal Response Byte method
15139                              <1>       ;          2 = Link IRQ by using Callback service method
15140                              <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
15141                              <1>       ;         >3 = invalid
15142                              <1>       ;
15143                              <1>       ;     CL = Signal Return/Response Byte value
```

```
15144                                  <1>       ;
15145                                  <1>       ;      If BH = 3, kernel will put a counter value  ; 03/08/2020
15146                                  <1>       ;                (into the S.R.B. addr)
15147                                  <1>       ;              between 0 to 255. (start value = CL+1)
15148                                  <1>       ;
15149                                  <1>       ;      NOTE: counter value, for example: even and odd numbers
15150                                  <1>       ;            may be used for -audio- DMA buffer switch
15151                                  <1>       ;            within double buffer method, etc.
15152                                  <1>       ;
15153                                  <1>       ;      EDX = Signal return (Response) byte address
15154                                  <1>       ;                           - or -
15155                                  <1>       ;            Interrupt/Callback service/routine address
15156                                  <1>       ;
15157                                  <1>       ;            (virtual address in user's memory space)
15158                                  <1>       ;
15159                                  <1>       ; OUTPUT ->
15160                                  <1>       ;      CF = 0 & EAX = 0 -> Successful setting
15161                                  <1>       ;      CF = 1 & EAX > 0 -> IRQ is prohibited or locked
15162                                  <1>       ;                   by another process
15163                                  <1>       ;              eax = ERR_PERM_DENIED -> prohibited or locked
15164                                  <1>       ;              eax = ERR_INV_PARAMETER ->
15165                                  <1>       ;                    invalid parameter/option or bad address
15166                                  <1>       ;
15167                                  <1>       ;      NOTE: Timer callbacks are set by using 'systimer'
15168                                  <1>       ;            system call (IRQ 0, PIT and IRQ 8, RTC)
15169                                  <1>       ;
15170                                  <1>       ;            Direct keyboard access is performed by using
15171                                  <1>       ;            Keyboard Interrupt (INT 32h)
15172                                  <1>       ;
15173                                  <1>       ;            It is prohibited here because:
15174                                  <1>       ;             1) Signal Response Byte method has not advantage
15175                                  <1>       ;                against INT 32h, function AH = 1. Also,
15176                                  <1>       ;                keyboard service interrupt will return with
15177                                  <1>       ;                ascii and scan codes (AL, AH) while
15178                                  <1>       ;                SRB method has only 1 byte space for ascii code
15179                                  <1>       ;                or scan code. One byte signal response is used
15180                                  <1>       ;                for ensuring very simple and very fast
15181                                  <1>       ;                virtual to physical memory address conversion
15182                                  <1>       ;                without any memory page crossover risk.
15183                                  <1>       ;                (Otherwise double page conversion or word
15184                                  <1>       ;                alignment would be needed.)
15185                                  <1>       ;             2) Badly written user code (callback code)
15186                                  <1>       ;                can prevent keyboard and timesharing functions
15187                                  <1>       ;                of the operating system via continuous and long
15188                                  <1>       ;                keyboard event handling by callback service.
15189                                  <1>       ;                (It can cause to lose immediate keystroke
15190                                  <1>       ;                response from hardware to user.)
15191                                  <1>       ;             3) If user will check any keyboard events, 'getkey'
15192                                  <1>       ;                (or 'getchar') must have more priority than other
15193                                  <1>       ;                (video etc.) events because only control ability
15194                                  <1>       ;                on a procedural infinite loop is a keyboard or
15195                                  <1>       ;                mouse event. So user can use keyboard function
15196                                  <1>       ;                at the end or at the beginning of a loop.
15197                                  <1>       ;                In this case, INT 32h is used for that purpose
15198                                  <1>       ;                and timer interrupt etc. callbacks can be used
15199                                  <1>       ;                for dynamic and synchronized data refresh/transfer
15200                                  <1>       ;                while cpu is in a static loop (without polling).
15201                                  <1>       ;                Keyboard Int callback is not more useful because
15202                                  <1>       ;                already a manual check (a key is pressed or not)
15203                                  <1>       ;                can be performed (via INT 32h, AH = 1) efficiently
15204                                  <1>       ;                in a loop to prevent a locked infinitive loop.
15205                                  <1>       ;
15206                                  <1>       ;          Disk IRQs (6,14,15) have been phohibited from ring 3
15207                                  <1>       ;          callback because, disk operations (file system services
15208                                  <1>       ;          etc.) are independent from user program, for fast disk r/w.
15209                                  <1>       ;          They are not more useful at ring 3 while they are in use
15210                                  <1>       ;          by standard diskio functions which are mandatory part of
15211                                  <1>       ;          (monolithic) OS kernel and mainprog command interpreter.
15212                                  <1>       ;          INT 33h diskio functions are enough for user level disk
15213                                  <1>       ;          r/w.
15214                                  <1>       ;
15215                                  <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
15216                                  <1>       ;
15217                                  <1>       ;         [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
15218                                  <1>       ;                   which IRQs are locked by (that) user.
15219                                  <1>       ;                    Lock and unlock (by user) will change
15220                                  <1>       ;                   these flags or 'terminate process' (sysexit)
15221                                  <1>       ;                   will clear these flags and unlock those IRQs.
15222                                  <1>       ;
15223                                  <1>       ;                   Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
15224                                  <1>       ;
15225                                  <1>       ;         IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
15226                                  <1>       ;
15227                                  <1>       ;         IRQ(x).method : 1 byte for callback method & status
15228                                  <1>       ;                       0 = Signal Response Byte method
15229                                  <1>       ;                       1 = Callback service method
15230                                  <1>       ;                       >1 = invalid for current 'syscalback'.
15231                                  <1>       ;                   or(+) 80h = IRQ is in use by system (ring 0)
15232                                  <1>       ;                               function (audio etc.) or
15233                                  <1>       ;                              a device driver.
15234                                  <1>       ;                   (system function will ignore the lock/owner)
15235                                  <1>       ;
15236                                  <1>       ;         IRQ(x).srb: 1 byte, Signal Return/Response byte value
15237                                  <1>       ;                    (a fixed value by user or a counter value
15238                                  <1>       ;                   from 0 to 255, which is increased by every
15239                                  <1>       ;                   interrupt just before putting it into
15240                                  <1>       ;                   the Signal Response byte address
15241                                  <1>       ;                   (This is not used in callback serv method)
15242                                  <1>       ;
15243                                  <1>       ;         IRQ(x).addr      : 1 dword
15244                                  <1>       ;                    Signal Response Byte address (physical)
15245                                  <1>       ;                              -or-
15246                                  <1>       ;                    Callback service address (virtual)
15247                                  <1>       ;
15248                                  <1>       ;         IRQ(x).dev: 1 byte
```

```
15249                                  <1>       ;                    0 = Default device or kernel function
15250                                  <1>       ;                                -or-
15251                                  <1>       ;                    1-255 = Assigned device driver number
15252                                  <1>       ;
15253                                  <1>       ;         (x) = 3,4,5,7,9,10,11,12,13
15254                                  <1>       ;
15255                                  <1>       ;
15256                                  <1>       ;      NOTE: If user's process/program calls the kernel (INT 40h)
15257                                  <1>       ;            while it is already running in a (ring 3) callback
15258                                  <1>       ;            service, kernel will force (convert) system call to
15259                                  <1>       ;            'sysrele' (sys release). So, this feature provides
15260                                  <1>       ;            easy and simple usage of callback services without
15261                                  <1>       ;            falling into deepless <please 'callback me' then
15262                                  <1>       ;            let me 'callback you'> cycles! (User must return
15263                                  <1>       ;            from callback service by using 'sysrele' system
15264                                  <1>       ;            call, without a significant delay. Otherwise user
15265                                  <1>       ;            process/program may be late to catch the next event
15266                                  <1>       ;            within same callback purpose.
15267                                  <1>       ;
15268                                  <1>
15269 00012415 30C0                   <1>       xor    al, al ; the caller is 'syscalbac' sign/flag
15270 00012417 E85F180000             <1>       call   set_irq_callback_service
15271                                  <1>       ; 16/04/2017
15272 0001241C A3[64030300]           <1>       mov    [u.r0], eax
15273 00012421 0F83E5B4FFFF           <1>       jnc    sysret
15274 00012427 A3[C8030300]           <1>       mov    dword [u.error], eax
15275 0001242C E9BBB4FFFF             <1>       jmp    error
15276                                  <1>
15277                                  <1> sysfpstat:
15278                                  <1>       ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
15279                                  <1>       ; (TRDOS 386 feature only!)
15280                                  <1>       ;
15281                                  <1>       ; Set or reset FPU registers save/restore option (for user)
15282                                  <1>       ;            (during software task switching, wswap-rswap)
15283                                  <1>       ;
15284                                  <1>       ; INPUT ->
15285                                  <1>       ;     BL = 0 -> reset
15286                                  <1>       ;     BL = 1 -> set (FPU register will be saved and restored)
15287                                  <1>       ;
15288                                  <1>       ; OUTPUT ->
15289                                  <1>       ;     cf = 0 -> no error, FPU is ready...
15290                                  <1>       ;               (EAX = 0)
15291                                  <1>       ;     Cf = 1 -> error, 80387 FPU is not ready !
15292                                  <1>       ;               (EAX = 0FFFFFFFFh)
15293                                  <1>
15294 00012431 31C0                   <1>       xor    eax, eax
15295 00012433 803D[D4960100]00       <1>       cmp    byte [fpready], 0
15296 0001243A 7613                   <1>       jna    short sysfpstat_err
15297                                  <1>
15298 0001243C 80E301                 <1>       and    bl, 1 ; use BIT 0 only !
15299 0001243F 881D[DA030300]         <1>       mov    [u.fpsave], bl
15300 00012445 A3[64030300]           <1>       mov    [u.r0], eax ; 0
15301 0001244A E9BDB4FFFF             <1>       jmp    sysret
15302                                  <1>
15303                                  <1> sysfpstat_err:
15304 0001244F 48                     <1>       dec    eax ; 0FFFFFFFFh
15305 00012450 A3[64030300]           <1>       mov    [u.r0], eax ; -1
15306 00012455 E992B4FFFF             <1>       jmp    error
15307                                  <1>
15308                                  <1> sysdelete: ; Delete (Remove, Unlink) File
15309                                  <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
15310                                  <1>       ;
15311                                  <1>         ; INPUT ->
15312                                  <1>         ;          EBX = File name (ASCIIZ string) address
15313                                  <1>       ; OUTPUT ->
15314                                  <1>       ;          cf = 0 -> eax = 0
15315                                  <1>       ;          cf = 1 -> Error code in AL
15316                                  <1>       ;
15317                                  <1>       ; Modified Registers: EAX (at the return of system call)
15318                                  <1>       ;
15319                                  <1>
15320 0001245A 89DE                   <1>       mov    esi, ebx
15321                                  <1>       ; file name is forced, change directory as temporary
15322                                  <1>       ;mov   ax, 1
15323                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
15324                                  <1>       ;call  set_working_path
15325 0001245C E8620B0000             <1>       call   set_working_path_x
15326 00012461 731D                   <1>       jnc    short sysdelete_1
15327                                  <1>
15328 00012463 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
15329 00012465 7505                   <1>       jnz    short sysdelete_err
15330                                  <1>       ; eax = 0
15331                                  <1> sysdelete_path_err:
15332 00012467 B813000000             <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
15333                                  <1> sysdelete_err:
15334 0001246C A3[64030300]           <1>       mov    [u.r0], eax
15335 00012471 A3[C8030300]           <1>       mov    [u.error], eax
15336 00012476 E81D0C0000             <1>       call   reset_working_path
15337 0001247B E96CB4FFFF             <1>       jmp    error
15338                                  <1> sysdelete_1:
15339                                  <1>       ;mov   esi, FindFile_Name
15340 00012480 66B80018               <1>       mov    ax, 1800h ; Only files
15341 00012484 E80870FFFF             <1>       call   find_first_file
15342 00012489 72E1                   <1>       jc     short sysdelete_err
15343                                  <1> sysdelete_2:
15344                                  <1>       ; check file attributes
15345                                  <1>
15346                                  <1>       ;test  bl, 17 ; system, hidden, readonly, directory
15347 0001248B F6C307                 <1>         test bl, 7 ; system, hidden, readonly
15348 0001248E 7407                   <1>         jz   short sysdelete_3
15349                                  <1>
15350 00012490 B80B000000             <1>         mov  eax, ERR_FILE_ACCESS ; 11 = 'permission denied !'
15351 00012495 EBD5                   <1>         jmp  short sysdelete_err
15352                                  <1> sysdelete_3:
15353 00012497 6621D2                 <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
```

```
15354 0001249A 7407                <1>       jz     short sysdelete_4
15355 0001249C B81A000000          <1>       mov    eax, ERR_INV_FILE_NAME ; 26 = 'invalid file name !'
15356 000124A1 EBC9                <1>         jmp  short sysdelete_err
15357                              <1> sysdelete_4:
15358                              <1>       ;mov   bh, [LongName_EntryLength]
15359 000124A3 883D[46940100]      <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
15360                              <1>       ; edi = Directory Entry Offset (DirBuff)
15361                              <1>       ; esi = Directory Entry (FFF Structure)
15362 000124A9 E86099FFFF         <1>       call   remove_file
15363 000124AE 72BC                <1>       jc     short sysdelete_err
15364                              <1> sysrmdir_5:
15365 000124B0 31C0                <1>       xor    eax, eax ; 0
15366 000124B2 A3[64030300]        <1>       mov    [u.r0], eax
15367                              <1>       ;mov   [u.error], eax
15368 000124B7 E8DC0B0000          <1>       call   reset_working_path
15369 000124BC E94BB4FFFF          <1>       jmp    sysret
15370                              <1>
15371                              <1>
15372                              <1> sysrmdir: ; Remove (Unlink) Directory
15373                              <1>       ; 19/01/2021
15374                              <1>       ; 29/12/2017 (TRDOS 386 = TRDOS v2.0)
15375                              <1>       ;
15376                              <1>         ; INPUT ->
15377                              <1>         ;          EBX = Pointer to directory name
15378                              <1>       ; OUTPUT ->
15379                              <1>       ;          cf = 0 -> eax = 0
15380                              <1>       ;          cf = 1 -> Error code in AL
15381                              <1>       ;
15382                              <1>       ; Modified Registers: EAX (at the return of system call)
15383                              <1>       ;
15384                              <1>
15385                              <1>       ; 19/01/2021
15386 000124C1 803D[B0030300]00    <1>       cmp    byte [u.uid], 0  ; root (super user) ?
15387 000124C8 7614                <1>       jna    short sysrmdir_0
15388                              <1>
15389                              <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
15390 000124CA B80B000000          <1>       mov    eax, ERR_PERM_DENIED ; ERR_NOT_SUPERUSER
15391 000124CF A3[64030300]        <1>       mov    [u.r0], eax
15392 000124D4 A3[C8030300]        <1>       mov    [u.error], eax
15393 000124D9 E90EB4FFFF          <1>       jmp    error
15394                              <1>
15395                              <1> sysrmdir_0:
15396 000124DE 89DE                <1>       mov    esi, ebx
15397                              <1>       ; file name is forced, change directory as temporary
15398                              <1>       ;mov   ax, 1
15399                              <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
15400                              <1>       ;call  set_working_path
15401 000124E0 E8DE0A0000          <1>       call   set_working_path_x
15402 000124E5 731D                <1>       jnc    short sysrmdir_1
15403                              <1>
15404 000124E7 21C0                <1>       and    eax, eax  ; 0 -> Bad Path!
15405 000124E9 7505                <1>       jnz    short sysrmdir_err
15406                              <1>       ; eax = 0
15407                              <1> sysrmdir_not_found:
15408 000124EB B80C000000          <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
15409                              <1> sysrmdir_err:
15410 000124F0 A3[64030300]        <1>       mov    [u.r0], eax
15411 000124F5 A3[C8030300]        <1>       mov    [u.error], eax
15412 000124FA E8990B0000          <1>       call   reset_working_path
15413 000124FF E9E8B3FFFF          <1>       jmp    error
15414                              <1> sysrmdir_1:
15415                              <1>       ;mov   esi, FindFile_Name
15416 00012504 66B81008            <1>       mov    ax, 0810h ; Only directories
15417 00012508 E8846FFFFF          <1>       call   find_first_file
15418 0001250D 7306                <1>       jnc    short sysrmdir_2
15419                              <1>
15420                              <1>       ; eax = 2 (File not found !)
15421 0001250F 3C02                <1>       cmp    al, 2 ; ERR_NOT_FOUND
15422 00012511 74D8                <1>       je     short sysrmdir_not_found
15423 00012513 EBDB                <1>       jmp    short sysrmdir_err
15424                              <1> sysrmdir_2:
15425                              <1>       ; check directory attributes
15426                              <1>
15427 00012515 F6C307              <1>         test bl, 7 ; system, hidden, readonly
15428 00012518 7407                <1>         jz   short sysrmdir_3
15429                              <1>
15430 0001251A B80B000000          <1>         mov  eax, ERR_DIR_ACCESS ; 11 = 'permission denied !'
15431 0001251F EBCF                <1>         jmp  short sysrmdir_err
15432                              <1> sysrmdir_3:
15433 00012521 6621D2              <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
15434 00012524 7407                <1>       jz     short sysrmdir_4
15435                              <1>       ;mov   eax, ERR_NOT_DIR ; 'not a valid directory !'
15436 00012526 B813000000          <1>       mov    eax, ERR_INV_PATH_NAME ; 'bad path name !'
15437 0001252B EBC3                <1>         jmp  short sysrmdir_err
15438                              <1> sysrmdir_4:
15439                              <1>       ;mov   bh, [LongName_EntryLength]
15440 0001252D 883D[46940100]      <1>       mov    [DelFile_LNEL], bh ; Long name entry length (if > 0)
15441                              <1>       ; edi = Directory Entry Offset (DirBuff)
15442                              <1>       ; esi = Directory Entry (FFF Structure)
15443 00012533 E82E76FFFF          <1>       call   delete_sub_directory
15444 00012538 0F8372FFFFFF        <1>       jnc    sysrmdir_5
15445                              <1> ;     jc     short sysrmdir_6
15446                              <1> ;
15447                              <1> ;     xor    eax, eax ; 0
15448                              <1> ;sysrmdir_5:
15449                              <1> ;     mov    [u.r0], eax
15450                              <1> ;     ;mov   [u.error], eax
15451                              <1> ;     call   reset_working_path
15452                              <1> ;     jmp    sysret
15453                              <1> sysrmdir_6:
15454 0001253E A3[64030300]        <1>       mov    [u.r0], eax
15455 00012543 A3[C8030300]        <1>       mov    [u.error], eax
15456                              <1>
15457 00012548 09C0                <1>       or     eax, eax ; EAX = 0 -> Directory not empty!
15458 0001254A 741C                <1>       jz     short sysrmdir_9
```

```
15459                                  <1>
15460                                  <1>       ; EAX > 0 -> Error code in AL (or AX or EAX)
15461                                  <1>
15462 0001254C 833D[FA910100]01        <1>       cmp    dword [FAT_ClusterCounter], 1
15463 00012553 7209                    <1>       jb     short sysrmdir_8
15464                                  <1> sysrmdir_7:
15465                                  <1>       ; ESI = Logical DOS Drive Description Table address
15466 00012555 66BB00FF                <1>       mov    bx, 0FF00h ; BH = FFh -> use ESI for Drive parameters
15467                                  <1>                  ; BL = 0 -> Recalculate free cluster count
15468 00012559 E894AEFFFF              <1>       call   calculate_fat_freespace
15469                                  <1> sysrmdir_8:
15470 0001255E E8350B0000              <1>       call   reset_working_path
15471 00012563 E984B3FFFF              <1>       jmp    error
15472                                  <1>
15473                                  <1> sysrmdir_9:
15474 00012568 A1[FA910100]            <1>       mov    eax, [FAT_ClusterCounter]
15475 0001256D 09C0                    <1>       or     eax, eax ; 0 ?
15476 0001256F 0F847BFFFFFF            <1>       jz     sysrmdir_err
15477                                  <1>       ; ESI = Logical DOS Drive Description Table address
15478 00012575 66BB01FF                <1>       mov    bx, 0FF01h ; BH = FFh -> use ESI for Drive parameters
15479                                  <1>                  ; BL = 1 -> add free clusters
15480 00012579 E874AEFFFF              <1>       call   calculate_fat_freespace
15481 0001257E 09C9                    <1>       or     ecx, ecx
15482 00012580 74DC                    <1>         jz   short sysrmdir_8 ; ecx = 0 -> OK
15483                                  <1>       ; ecx > 0 -> Error (Recalculation is needed)
15484 00012582 EBD1                    <1>       jmp    short sysrmdir_7
15485                                  <1>
15486                                  <1>
15487                                  <1> syschdir: ; Change Current (Working) Drive & Directory (for user)
15488                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15489                                  <1>       ;
15490                                  <1>         ; INPUT ->
15491                                  <1>         ;          EBX = Directory name (ASCIIZ string) address
15492                                  <1>       ; OUTPUT ->
15493                                  <1>       ;          cf = 0 -> eax = 0
15494                                  <1>       ;          cf = 1 -> Error code in AL
15495                                  <1>       ;
15496                                  <1>       ; Modified Registers: EAX (at the return of system call)
15497                                  <1>       ;
15498                                  <1>       ; NOTE: If drive name is not included, only the working
15499                                  <1>       ; directory (for user, not for drive/OS) will be chanded.
15500                                  <1>       ; If there is a drive name (as A:, B:, C:, D: etc.)
15501                                  <1>       ; at the beginning of the ASCIIZ (directory) string,
15502                                  <1>       ; working drive and working directory (for user)
15503                                  <1>       ; will be changed together.
15504                                  <1>       ; (When the program is terminated, MainProg -internal
15505                                  <1>       ; shell- will reset working directory to the previous
15506                                  <1>       ; -current- logical drive's current directory again.)
15507                                  <1>
15508 00012584 89DE                    <1>       mov    esi, ebx
15509                                  <1>       ; file name is not forced, change directory as temporary
15510 00012586 31C0                    <1>       xor    eax, eax
15511                                  <1>       ;mov   [FFF_Valid], ah ; 0 ; reset
15512                                  <1>       ;call  set_working_path
15513 00012588 E83A0A0000              <1>       call   set_working_path_xx
15514 0001258D 731D                    <1>       jnc    short syschdir_ok
15515 0001258F 21C0                    <1>       and    eax, eax  ; 0 -> Bad Path!
15516 00012591 7505                    <1>       jnz    short syschdir_err
15517                                  <1>       ; eax = 0
15518                                  <1> syschdir_not_found:
15519 00012593 B80C000000              <1>       mov    eax, ERR_DIR_NOT_FOUND ; Directory not found !
15520                                  <1> syschdir_err:
15521 00012598 A3[64030300]            <1>       mov    [u.r0], eax
15522 0001259D A3[C8030300]            <1>       mov    [u.error], eax
15523 000125A2 E8F10A0000              <1>       call   reset_working_path
15524 000125A7 E940B3FFFF              <1>       jmp    error
15525                                  <1> syschdir_ok:
15526 000125AC 31C0                    <1>       xor    eax, eax ; 0
15527 000125AE A3[64030300]            <1>       mov    [u.r0], eax
15528                                  <1>       ;mov   [u.error], eax
15529 000125B3 E954B3FFFF              <1>       jmp    sysret
15530                                  <1>
15531                                  <1>
15532                                  <1> syschmod: ; Get & Change File (or Directory) Attributes
15533                                  <1>       ; 19/01/2021
15534                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15535                                  <1>       ;
15536                                  <1>         ; INPUT ->
15537                                  <1>         ;          EBX = File/Directory (ASCIIZ) name address
15538                                  <1>       ;          CL = New attributes (if CL < 40h)
15539                                  <1>       ;          CL >= 40h -> Get File Attributes
15540                                  <1>       ; OUTPUT ->
15541                                  <1>       ;          cf = 0 -> EAX = File attributes (in AL)
15542                                  <1>       ;          cf = 1 -> Error code in AL
15543                                  <1>       ;
15544                                  <1>       ; Modified Registers: EAX (at the return of system call)
15545                                  <1>       ;
15546                                  <1>       ; MSDOS File Attributes:    (bit value of attrib byte)
15547                                  <1>       ;     ATTR_READ_ONLY      =      01h  (bit 0, 'R')
15548                                  <1>       ;     ATTR_HIDDEN  =      02h  (bit 1, 'H')
15549                                  <1>       ;     ATTR_SYSTEM  =      04h  (bit 2, 'S')
15550                                  <1>       ;     ATTR_VOLUME_ID      =      08h  (bit 3)
15551                                  <1>       ;     ATTR_DIRECTORY      =      10h  (bit 4)
15552                                  <1>       ;     ATTR_ARCHIVE =      20h  (bit 5, 'A')
15553                                  <1>       ;     ATTR_LONG_NAME      =      ATTR_READONLY |
15554                                  <1>       ;                          ATTR_HIDDEN |
15555                                  <1>       ;                          ATTR_SYSTEM |
15556                                  <1>       ;                          ATTR_VOLUME_ID
15557                                  <1>       ;     The upper two bits of attributes must be 0.
15558                                  <1>
15559                                  <1>       ; Note:      * If ATTR_DIRECTORY is set, only directory names
15560                                  <1>       ;       will be searched (and S,H,R,A attributeds of
15561                                  <1>       ;       the directory will be changed.)
15562                                  <1>       ;     * If ATTR_VOLUME_ID is set, 'syschmod' system call
15563                                  <1>       ;       will return with 'permission denied' error.
```

```
15564                                  <1>       ;      * If ATTR_DIRECTORY is not set, only file names
15565                                  <1>       ;        will be searched (and S,H,R,A attributes of the
15566                                  <1>       ;        file will be changed.)
15567                                  <1>       ;
15568                                  <1>       ; (Ony Super User can change S,H,R attributes.)
15569                                  <1>
15570 000125B8 80F940                 <1>       cmp    cl, 40h
15571 000125BB 7327                   <1>       jnb    short syschmod_0
15572                                  <1>
15573 000125BD F6C108                 <1>       test   cl, 08h ; ATTR_VOLUME_ID
15574 000125C0 750E                   <1>       jnz    short syschmod_perm_err
15575                                  <1>
15576                                  <1>       ; 19/01/2021
15577 000125C2 803D[B0030300]00       <1>       cmp    byte [u.uid], 0  ; root (super user) ?
15578 000125C9 7619                   <1>       jna    short syschmod_0
15579                                  <1>
15580                                  <1>       ; Not super user..
15581 000125CB F6C107                 <1>       test   cl, 07h      ; S,H,R attributes
15582 000125CE 7414                   <1>       jz     short syschmod_0
15583                                  <1>
15584                                  <1> syschmod_perm_err:
15585                                  <1>       ;mov   dword [u.r0], ERR_PERM_DENIED
15586 000125D0 B80B000000             <1>       mov    eax, ERR_PERM_DENIED ; 'permission denied !'
15587 000125D5 A3[64030300]           <1>       mov    [u.r0], eax
15588 000125DA A3[C8030300]           <1>       mov    [u.error], eax
15589 000125DF E908B3FFFF             <1>       jmp    error
15590                                  <1>
15591                                  <1> syschmod_0:
15592 000125E4 880D[94940100]         <1>       mov    [Attributes], cl
15593 000125EA 89DE                   <1>       mov    esi, ebx
15594                                  <1>       ; file name is forced, change directory as temporary
15595                                  <1>       ;mov   ax, 1
15596                                  <1>       ;mov   [FFF Valid], ah ; 0 ; reset
15597                                  <1>       ;call  set_working_path
15598 000125EC E8D2090000             <1>       call   set_working_path_x
15599 000125F1 731D                   <1>       jnc    short syschmod_1
15600 000125F3 21C0                   <1>       and    eax, eax  ; 0 -> Bad Path!
15601 000125F5 7505                   <1>       jnz    short syschmod_err
15602                                  <1>       ; eax = 0
15603                                  <1> syschmod_path_not_found:
15604 000125F7 B813000000             <1>       mov    eax, ERR_INV_PATH_NAME ; 'Bad path name !'
15605                                  <1> syschmod_err:
15606 000125FC A3[64030300]           <1>       mov    [u.r0], eax
15607 00012601 A3[C8030300]           <1>       mov    [u.error], eax
15608 00012606 E88D0A0000             <1>       call   reset_working_path
15609 0001260B E9DCB2FFFF             <1>       jmp    error
15610                                  <1> syschmod_1:
15611 00012610 B008                   <1>       mov    al, 08h ; Except volume labels (& long names)
15612 00012612 A0[94940100]           <1>       mov    al, [Attributes]
15613 00012617 2410                   <1>       and    al, 10h ;
15614                                  <1>       ;mov   esi, FindFile_Name
15615                                  <1>       ;mov   ax, 1800h ; Only files
15616                                  <1>       ;mov   ax, 0810h ; Only directories
15617 00012619 E8736EFFFF             <1>       call   find_first_file
15618                                  <1>       ;jnc   short syschmod_2
15619 0001261E 72DC                   <1>       jc     short syschmod_err
15620                                  <1>
15621                                  <1>       ;; eax = 2 (File not found !)
15622                                  <1>       ;cmp   al, 2 ; ERR_NOT_FOUND
15623                                  <1>       ;jne   short syschmod_err
15624                                  <1>
15625                                  <1>       ;and   byte [Attributes], 10h
15626                                  <1>       ;jz    short syschmod_err
15627                                  <1>
15628                                  <1>       ;; Directory not found !
15629                                  <1>       ;mov   al, 3 ; ERR_PATH_NOT_FOUND
15630                                  <1>       ;jmp   short syschmod_err
15631                                  <1>
15632                                  <1> syschmod_2:
15633 00012620 6621D2                 <1>       and    dx, dx ; Ambiguous filename chars used sign (DX>0)
15634 00012623 7407                   <1>       jz     short syschmod_3
15635 00012625 B81A000000             <1>       mov    eax, ERR_INV_FILE_NAME ; 'invalid file name !'
15636 0001262A EBD0                   <1>         jmp  short syschmod_err
15637                                  <1> syschmod_3:
15638                                  <1>       ; EDI = Directory buffer entry offset/address
15639                                  <1>       ; BL = File (or Directory) Attributes
15640                                  <1>       ; mov  bl, [EDI+0Bh]
15641                                  <1>
15642                                  <1>       ; check directory attributes
15643 0001262C 8A3D[94940100]         <1>       mov    bh, [Attributes] ; new attributes
15644 00012632 80FF40                 <1>       cmp    bh, 40h  ;>=40 -> get file/directory attributes
15645 00012635 732D                   <1>       jnb    short syschmod_6
15646                                  <1>
15647                                  <1>       ; set file/directory attributes
15648 00012637 F6C307                 <1>       test   bl, 7 ; system, hidden, readonly
15649 0001263A 7409                   <1>         jz   short syschmod_4
15650                                  <1>
15651                                  <1>       ; 19/01/2021
15652 0001263C 803D[B0030300]00       <1>       cmp    byte [u.uid], 0  ; root (super user) ?
15653 00012643 778B                   <1>       ja     short syschmod_perm_err
15654                                  <1> syschmod_4:
15655 00012645 66817F0CA101           <1>       cmp    word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
15656 0001264B 7424                   <1>       je     short syschmod_7
15657                                  <1>
15658 0001264D 887F0B                 <1>       mov    [edi+0Bh], bh    ; Attributes (New!)
15659                                  <1>
15660 00012650 C605[04920100]02       <1>       mov    byte [DirBuff_ValidData], 2 ; modified sign
15661                                  <1>                                       ; to force write
15662 00012657 E86A94FFFF             <1>       call   save_directory_buffer
15663 0001265C 729E                   <1>       jc     short syschmod_err
15664                                  <1>
15665                                  <1> syschmod_5:
15666 0001265E 8A1D[94940100]         <1>       mov    bl, [Attributes]
15667                                  <1> syschmod_6:
15668 00012664 0FB6C3                 <1>       movzx  eax, bl
```

```
15669 00012667 A3[64030300]        <1>       mov    [u.r0], eax
15670                              <1>       ;mov   dword [u.error], 0
15671 0001266C E99BB2FFFF          <1>       jmp    sysret
15672                              <1>
15673                              <1> syschmod_7:
15674 00012671 29C0                <1>       sub    eax, eax
15675 00012673 8A25[02920100]      <1>         mov     ah, [DirBuff_DRV]
15676 00012679 BE00010900          <1>       mov    esi, Logical_DOSDisks
15677 0001267E 01C6                <1>         add     esi, eax
15678 00012680 807E04A1            <1>         cmp     byte [esi+LD_FSType], 0A1h
15679 00012684 7307                <1>       jnc    short syschmod_8
15680 00012686 B01D                <1>       mov    al, ERR_INV_DATA ; 29 = Invalid Data
15681 00012688 E96FFFFFFF          <1>       jmp    syschmod_err
15682                              <1>
15683                              <1> syschmod_8:
15684                              <1>       ; BH = New MS-DOS File Attributes
15685 0001268D 88F8                <1>       mov    al, bh ; File/Directory Attributes
15686 0001268F 30E4                <1>       xor    ah, ah ; Attributes in MS-DOS format sign
15687 00012691 E8597FFFFF          <1>       call   change_fs_file_attributes
15688 00012696 0F8260FFFFFF        <1>       jc     syschmod_err
15689 0001269C EBC0                <1>       jmp    short syschmod_5
15690                              <1>
15691                              <1>
15692                              <1> sysdrive: ; Get/Set Current (Working) Drive (for user)
15693                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15694                              <1>       ;
15695                              <1>         ; INPUT ->
15696                              <1>         ;          BL = Logical DOS Drive number (0=A: ... 2=C:)
15697                              <1>       ;         If BL = 0FFh -> Get Current Drive
15698                              <1>       ; OUTPUT ->
15699                              <1>       ;          cf = 0 ->
15700                              <1>       ;               AL = Current Drive number
15701                              <1>       ;               AH = The Last Logical DOS Drive no.
15702                              <1>       ;          cf = 1 -> Error code in AL
15703                              <1>       ;
15704                              <1>       ; Modified Registers: EAX (at the return of system call)
15705                              <1>       ;
15706                              <1>       ; NOTE: If the requested logical dos drive is ready,
15707                              <1>       ;     it's current current directory will be the user's
15708                              <1>       ;     (program's) current directory.
15709                              <1>       ;     (When the program is terminated, MainProg -internal
15710                              <1>       ;     shell- will reset the previous -current- logical drive
15711                              <1>       ;       as current drive again).
15712                              <1>
15713 0001269E 80FBFF              <1>       cmp    bl, 0FFh
15714 000126A1 7435                <1>       je     short sysdrive_ok
15715 000126A3 3A1D[A5400100]      <1>       cmp    bl, [Last_DOS_DiskNo]
15716 000126A9 771E                <1>       ja     short sysdrive_err
15717                              <1>
15718                              <1>       ; Save current drive and reset mode
15719                              <1>       ; for 'reset_working_path' procedure (for MainProg)
15720 000126AB 30C0                <1>       xor    al, al
15721 000126AD 66A3[D0960100]      <1>       mov    [SWP_Mode], ax ; ah = 0
15722 000126B3 A0[DE8A0100]        <1>       mov    al, [Current_Drv]
15723 000126B8 FEC4                <1>       inc    ah ; mov ah, 1
15724 000126BA 66A3[D2960100]      <1>       mov    [SWP_DRV], ax
15725                              <1>
15726 000126C0 88DA                <1>       mov    dl, bl
15727 000126C2 E8275AFFFF          <1>       call   change_current_drive
15728 000126C7 730F                <1>       jnc    short sysdrive_ok
15729                              <1> sysdrive_err:
15730 000126C9 C705[64030300]0F00- <1>       mov    dword [u.r0], ERR_DRV_NOT_RDY ; 'drive not ready !'
15730 000126D1 0000                <1>
15731 000126D3 E914B2FFFF          <1>       jmp    error
15732                              <1> sysdrive_ok:
15733 000126D8 A0[DE8A0100]        <1>       mov    al, [Current_Drv]
15734 000126DD 8A25[A5400100]      <1>       mov    ah, [Last_DOS_DiskNo]
15735 000126E3 A3[64030300]        <1>       mov    [u.r0], eax
15736 000126E8 E91FB2FFFF          <1>       jmp    sysret
15737                              <1>
15738                              <1>
15739                              <1> sysdir: ; Get Current (Working) Drive & Directory (for user)
15740                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15741                              <1>       ;
15742                              <1>         ; INPUT ->
15743                              <1>         ;          EBX = Current directory name buffer address
15744                              <1>       ;            (Buffer length = 92 bytes)
15745                              <1>       ; OUTPUT ->
15746                              <1>       ;          AL = Current drive (0=A: .. 2=C:)
15747                              <1>       ;         If CF = 1 -> AL = error code
15748                              <1>       ;
15749                              <1>       ; Modified Registers: EAX (at the return of system call)
15750                              <1>       ;
15751                              <1>       ; Note: Required directory name buffer length may be
15752                              <1>       ;     <= 92 bytes for current TRDOS 386 version.
15753                              <1>       ;     (7*12 name chars + 7 slash + 0)
15754                              <1>
15755 000126ED 89E5                <1>       mov    ebp, esp
15756 000126EF 83EC60              <1>       sub    esp, 96
15757 000126F2 53                  <1>       push   ebx ; User's buffer address
15758 000126F3 30D2                <1>       xor    dl, dl ; 0 = current drive
15759 000126F5 E8F088FFFF          <1>       call   get_current_directory
15760 000126FA 72CD                <1>       jc     short sysdrive_err ; 'drive not ready !' error
15761 000126FC 89E6                <1>       mov    esi, esp ; System's buffer address
15762 000126FE 5F                  <1>       pop    edi  ; User's buffer address
15763                              <1>       ; ecx = transfer (byte) count (<=92)
15764 000126FF E843F4FFFF          <1>       call   transfer_to_user_buffer
15765 00012704 89EC                <1>       mov    esp, ebp
15766 00012706 730F                <1>       jnc    short sysdir_ok
15767                              <1> sysdir_err:
15768 00012708 C705[64030300]2E00- <1>       mov    dword [u.r0], ERR_BUFFER  ; 'buffer error !'
15768 00012710 0000                <1>
15769 00012712 E9D5B1FFFF          <1>       jmp    error
15770                              <1> sysdir_ok:
15771 00012717 8A0D[DE8A0100]      <1>       mov    cl, [Current_Drv]
```

```
15772 0001271D 890D[64030300]      <1>       mov    [u.r0], ecx
15773 00012723 E9E4B1FFFF          <1>       jmp    sysret
15774                              <1>
15775                              <1>
15776                              <1> sysldrvt: ; Get copy of Logical DOS Drive Description Table
15777                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15778                              <1>       ;
15779                              <1>         ; INPUT ->
15780                              <1>       ;          BL = Logical DOS drive number (zero based)
15781                              <1>         ;          ECX = Logical DOS drv desc table buffer addr
15782                              <1>       ;            (Buffer length = 256 bytes)
15783                              <1>       ; OUTPUT ->
15784                              <1>       ;          cf = 0 ->
15785                              <1>       ;               AL = Current Drive number
15786                              <1>       ;               AH = The Last Logical DOS Drive no.
15787                              <1>       ;          cf = 1 -> Error code in AL
15788                              <1>       ;               AH = The Last Logical DOS Drive no.
15789                              <1>       ;
15790                              <1>       ; Modified Registers: EAX (at the return of system call)
15791                              <1>       ;
15792                              <1>       ; Note: Required description table buffer length is
15793                              <1>       ;     256 bytes for current TRDOS 386 version.
15794                              <1>
15795 00012728 89CF               <1>       mov    edi, ecx ; Destination address (user space)
15796 0001272A 88DC               <1>       mov    ah, bl
15797 0001272C 30C0               <1>       xor    al, al
15798 0001272E BE00010900         <1>       mov    esi, Logical_DOSDisks
15799 00012733 01C6               <1>       add    esi, eax ; Source address (system space)
15800 00012735 B900010000         <1>       mov    ecx, 256 ; Byte count
15801                              <1>                       ; Logical Dos Drv Desc Table size
15802 0001273A E808F4FFFF         <1>       call   transfer_to_user_buffer
15803 0001273F 72C7               <1>       jc     short sysdir_err
15804 00012741 8A2D[A5400100]     <1>       mov    ch, [Last_DOS_DiskNo]
15805 00012747 EBCE               <1>       jmp    short sysdir_ok
15806                              <1>
15807                              <1>
15808                              <1> systime: ; Get System Date&Time
15809                              <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15810                              <1>       ;
15811                              <1>         ; INPUT -> BL =
15812                              <1>       ;          0 = Get Date&Time in Unix/Epoch format
15813                              <1>       ;          1 = Get Time in MSDOS format
15814                              <1>       ;          2 = Get Date in MSDOS format
15815                              <1>       ;          3 = Get Date&Time in MSDOS format
15816                              <1>       ;          4 & other values =
15817                              <1>       ;             System timer ticks will be returned
15818                              <1>       ;             in EAX and Carry Flag will be set.
15819                              <1>       ;             (CF will not be set if BL = 4)
15820                              <1>       ; OUTPUT ->
15821                              <1>       ;      For BL input = 3
15822                              <1>       ;           EAX = Current Time (RTC)
15823                              <1>       ;             AL = Second (DL in MSDOS)
15824                              <1>       ;             AH = Minute (CL in MSDOS)
15825                              <1>       ;             HW of EAX = Hour (CH in MSDOS)
15826                              <1>       ;         EDX = Current System Date (RTC)
15827                              <1>       ;             DL = Day (DL in MSDOS)
15828                              <1>       ;             DH = Month (DH in MSDOS)
15829                              <1>       ;             HW of EDX = Year (CX in MSDOS)
15830                              <1>       ;
15831                              <1>       ;      For BL input = 2
15832                              <1>       ;         EAX = Current System Date (RTC)
15833                              <1>       ;             DL = Day (DL in MSDOS)
15834                              <1>       ;             DH = Month (DH in MSDOS)
15835                              <1>       ;             HW of EDX = Year (CX in MSDOS)
15836                              <1>       ;
15837                              <1>       ;      For BL input = 1
15838                              <1>       ;           EAX = Current Time (RTC)
15839                              <1>       ;             AL = Second (DL in MSDOS)
15840                              <1>       ;             AH = Minute (CL in MSDOS)
15841                              <1>       ;             HW of EAX = Hour (CH in MSDOS)
15842                              <1>       ;
15843                              <1>       ;      For BL input = 0
15844                              <1>       ;           EAX = Unix (Epoch) Time Ticks/Seconds
15845                              <1>       ;
15846                              <1>       ;      For BL input  = 4
15847                              <1>       ;         EAX = System timer ticks
15848                              <1>       ;
15849                              <1>       ;      If CF = 1 (for other values of BL input)
15850                              <1>       ;         EAX = System timer ticks (no error code!)
15851                              <1>       ;
15852                              <1>       ; Modified Registers: EAX, (EDX)
15853                              <1>       ;              (at the return of system call)
15854                              <1>       ;
15855                              <1>
15856 00012749 20DB               <1>       and    bl, bl
15857 0001274B 750F               <1>       jnz    short systime_1
15858 0001274D E88C4FFFFF         <1>       call   epoch
15859                              <1> systime_0:
15860 00012752 A3[64030300]       <1>       mov    [u.r0], eax
15861 00012757 E9B0B1FFFF         <1>       jmp    sysret
15862                              <1> systime_1:
15863 0001275C 80FB04             <1>       cmp    bl, 4
15864 0001275F 7211               <1>       jb     short systime_2
15865 00012761 A1[988A0100]       <1>       mov    eax, [TIMER_LH] ; 18.2 Hz timer ticks
15866                              <1>                             ; Note: [TIMER_LH] may be set
15867                              <1>                             ; to wrong timer value due to
15868                              <1>                             ; program functions.
15869                              <1>                             ; (This value must not be
15870                              <1>                             ; accepted as [TIMER_LH]/18.2
15871                              <1>                             ; seconds since the midnight.)
15872 00012766 76EA               <1>       jna    short systime_0
15873 00012768 A3[64030300]       <1>       mov    [u.r0], eax
15874 0001276D E97AB1FFFF         <1>       jmp    error ; cf = 1 & [u.r0] = eax = timer ticks
15875                              <1>
15876                              <1> systime_2:
```

```
15877                                  <1>       ;push  ebx
15878 00012772 E8C94EFFFF              <1>       call   get_rtc_date_time
15879                                  <1>       ;pop   ebx
15880 00012777 F6C301                  <1>       test   bl, 1
15881 0001277A 7429                    <1>       jz     short systime_4
15882 0001277C 30E4                    <1>       xor    ah, ah
15883 0001277E A0[E2860100]            <1>       mov    al, [hour]
15884 00012783 88C2                    <1>       mov    dl, al
15885 00012785 C1E010                  <1>       shl    eax, 16
15886 00012788 A0[E6860100]            <1>       mov    al, [second]
15887 0001278D 8A25[E4860100]          <1>       mov    ah, [minute]
15888 00012793 F6C302                  <1>       test   bl, 2
15889 00012796 74BA                    <1>       jz     short systime_0
15890                                  <1>       ; Check time & date match risk
15891                                  <1>       ; (23:59:59 may cause to wrong
15892                                  <1>       ; date -new day with previous date-...)
15893 00012798 80FA17                  <1>       cmp    dl, 23
15894 0001279B 7206                    <1>       jb     short systime_3
15895 0001279D 663D3B3B                <1>       cmp    ax, (59*256)+59 ; if hour is 23:59:59
15896 000127A1 73CF                    <1>       jnb    short systime_2 ; wait for 1 second
15897                                  <1> systime_3:
15898                                  <1>       ; eax = time
15899 000127A3 89C6                    <1>       mov    esi, eax
15900                                  <1> systime_4:
15901 000127A5 66A1[DC860100]          <1>       mov    ax, [year]
15902 000127AB C1E010                  <1>       shl    eax, 16
15903 000127AE A0[E0860100]            <1>       mov    al, [day]
15904 000127B3 8A25[DE860100]          <1>       mov    ah, [month]
15905                                  <1>       ; eax = date
15906 000127B9 80E301                  <1>       and    bl, 1
15907 000127BC 7494                    <1>       jz     short systime_0
15908 000127BE 96                      <1>       xchg   esi, eax
15909                                  <1>       ; eax = time, esi = date
15910 000127BF 8B2D[60030300]          <1>       mov    ebp, [u.usp]  ; EBP points to user's registers
15911                                  <1>       ; (user) edx <-- (system) esi
15912 000127C5 897514                  <1>       mov    [ebp+20], esi ; return to user with EDX value
15913 000127C8 EB88                    <1>       jmp    short systime_0
15914                                  <1>
15915                                  <1>
15916                                  <1> sysstime: ; Set System Date&Time
15917                                  <1>       ; 31/12/2017
15918                                  <1>       ; 30/12/2017 (TRDOS 386 = TRDOS v2.0)
15919                                  <1>       ;
15920                                  <1>         ; INPUT -> BL =
15921                                  <1>       ;          0 = Set Date&Time in Unix/Epoch format
15922                                  <1>       ;          1 = Set Time in MSDOS format
15923                                  <1>       ;          2 = Set Date in MSDOS format
15924                                  <1>       ;          3 = Set Date&Time in MSDOS format
15925                                  <1>       ;          4 = Set System Timer (Ticks)
15926                                  <1>       ;          5 = Convert/Save current time to/as
15927                                  <1>       ;             18.2 Hz system timer ticks
15928                                  <1>       ;          6 = Convert MSDOS Date&Time to UNIX format
15929                                  <1>       ;             without setting system date&time ; (test)
15930                                  <1>       ;          7 = Convert UNIX Date&Time to MSDOS format
15931                                  <1>       ;             without setting system date&time ; (test)
15932                                  <1>       ;         8-0FFh = invalid !
15933                                  <1>       ;        ECX = Time (or Timer) value in selected format
15934                                  <1>       ;        EDX = Date value in MSDOS format if BL=2,3,6
15935                                  <1>       ;
15936                                  <1>       ; OUTPUT ->
15937                                  <1>       ;      If CF = 0 ->
15938                                  <1>       ;           EAX = Set value
15939                                  <1>       ;      If CF = 1 -> (invalid BL input)
15940                                  <1>       ;         EAX = Ticks count [TIMER_LH]
15941                                  <1>       ;
15942                                  <1>
15943 000127CA 20DB                    <1>       and    bl, bl ; 0
15944 000127CC 7511                    <1>       jnz    short sysstime_0
15945 000127CE 89C8                    <1>       mov    eax, ecx
15946 000127D0 E8934FFFFF              <1>       call   convert_from_epoch
15947 000127D5 E83F50FFFF              <1>       call   set_rtc_date_time
15948 000127DA E92DB1FFFF              <1>       jmp    sysret
15949                                  <1> sysstime_0:
15950 000127DF 80FB08                  <1>       cmp    bl, 8
15951 000127E2 722D                    <1>       jb     short sysstime_1
15952                                  <1>       ; invalid input (>7)
15953 000127E4 A1[988A0100]            <1>       mov    eax, [TIMER_LH] ; 18.2 Hz timer ticks
15954                                  <1>                               ; Note: [TIMER_LH] may be set
15955                                  <1>                               ; to wrong timer value due to
15956                                  <1>                               ; program functions.
15957                                  <1>                               ; (This value must not be
15958                                  <1>                               ; accepted as [TIMER_LH]/18.2
15959                                  <1>                               ; seconds since the midnight.)
15960 000127E9 A3[64030300]            <1>       mov    [u.r0], eax
15961 000127EE E9F9B0FFFF              <1>       jmp    error ; cf = 1 & [u.r0] = eax = timer ticks
15962                                  <1>
15963                                  <1> sysstime_8:
15964                                  <1>       ; BL = 7
15965 000127F3 89C8                    <1>       mov    eax, ecx ; seconds since 1/1/1970 00:00:00
15966 000127F5 E86E4FFFFF              <1>       call   convert_from_epoch
15967 000127FA 30E4                    <1>       xor    ah, ah
15968 000127FC A0[E2860100]            <1>       mov    al, [hour]
15969 00012801 C1E010                  <1>       shl    eax, 16
15970 00012804 A0[E6860100]            <1>       mov    al, [second]
15971 00012809 8A25[E4860100]          <1>       mov    ah, [minute]
15972 0001280F EB92                    <1>       jmp    short systime_3
15973                                  <1>
15974                                  <1> sysstime_1:
15975 00012811 80FB04                  <1>       cmp    bl, 4
15976 00012814 743F                    <1>       je     short sysstime_2 ; set system timer ticks
15977 00012816 80FB05                  <1>       cmp    bl, 5
15978 00012819 754B                    <1>       jne    short sysstime_4
15979                                  <1>       ; convert current time to system timer ticks (18.2Hz)
15980 0001281B E8204EFFFF              <1>       call   get_rtc_date_time
15981 00012820 0FB60D[E2860100]        <1>       movzx  ecx, byte [hour]
```

```
15982 00012827 B8100E0000          <1>       mov   eax, 60*60 ; 1 hour = 3600 seconds
15983 0001282C F7E1                <1>       mul   ecx
15984 0001282E 89C3                <1>       mov   ebx, eax
15985 00012830 B13C                <1>       mov   cl, 60  ; 1 minute = 60 seconds
15986 00012832 0FB605[E4860100]    <1>       movzx eax, byte [minute]
15987 00012839 F7E1                <1>       mul   ecx
15988 0001283B 01D8                <1>       add   eax, ebx
15989 0001283D 8A0D[E6860100]      <1>       mov   cl, [second]
15990 00012843 01C8                <1>       add   eax, ecx
15991 00012845 B1B6                <1>       mov   cl, 182
15992 00012847 F7E1                <1>       mul   ecx
15993 00012849 83C009              <1>       add   eax, 9
15994 0001284C 83D200              <1>       adc   edx, 0
15995 0001284F B10A                <1>       mov   cl, 10
15996 00012851 F7F1                <1>       div   ecx
15997                              <1>       ; eax = ((182*seconds)+9)/10
15998 00012853 89C1                <1>       mov   ecx, eax
15999                              <1> sysstime_2:
16000 00012855 890D[988A0100]      <1>       mov   [TIMER_LH], ecx ; 18.2 * seconds
16001                              <1> sysstime_3:
16002 0001285B 890D[64030300]      <1>       mov   [u.r0], ecx
16003 00012861 E9A6B0FFFF          <1>       jmp   sysret
16004                              <1> sysstime_4:
16005 00012866 80FB06              <1>       cmp   bl, 6
16006 00012869 7788                <1>       ja    short sysstime_8
16007                              <1>
16008 0001286B 890D[64030300]      <1>       mov   [u.r0], ecx
16009                              <1>
16010 00012871 880D[E6860100]      <1>       mov   [second], cl
16011 00012877 882D[E4860100]      <1>       mov   [minute], ch
16012 0001287D C1E910              <1>       shr   ecx, 16
16013 00012880 880D[E2860100]      <1>       mov   [hour], cl
16014                              <1>       ; BL = 1,2,3,6
16015 00012886 80FB01              <1>       cmp   bl, 1
16016 00012889 762A                <1>       jna   short sysstime_5
16017                              <1>       ; BL = 2,3,6
16018 0001288B 8815[E0860100]      <1>       mov   [day], dl
16019 00012891 8835[DE860100]      <1>       mov   [month], dh
16020 00012897 C1EA10              <1>       shr   edx, 16
16021 0001289A 668915[DC860100]    <1>       mov   [year], dx
16022 000128A1 80E303              <1>       and   bl, 3
16023 000128A4 742D                <1>       jz    short sysstime_7 ; 6
16024                              <1>       ; BL = 2,3
16025 000128A6 F6C301              <1>       test  bl, 1
16026 000128A9 7419                <1>       jz    short sysstime_6 ; 2
16027                              <1>       ; BL = 3
16028 000128AB E8694FFFFF          <1>       call  set_rtc_date_time
16029 000128B0 E957B0FFFF          <1>       jmp   sysret
16030                              <1> sysstime_5:
16031                              <1>       ; BL = 1
16032 000128B5 E8A04FFFFF          <1>       call  set_time_bcd
16033 000128BA E82242FFFF          <1>       call  set_rtc_time
16034 000128BF E948B0FFFF          <1>       jmp   sysret
16035                              <1> sysstime_6:
16036                              <1>       ; BL = 2
16037 000128C4 E8644FFFFF          <1>       call  set_date_bcd
16038 000128C9 E88242FFFF          <1>       call  set_rtc_date
16039 000128CE E939B0FFFF          <1>       jmp   sysret
16040                              <1> sysstime_7:
16041                              <1>       ; BL = 6
16042                              <1>       ; [year], [month], [day],
16043                              <1>       ; [hour], [minute], [second]
16044 000128D3 E80B4EFFFF          <1>       call  convert_to_epoch
16045 000128D8 89C1                <1>       mov   ecx, eax ; seconds since 1/1/1970 00:00:00
16046 000128DA E97CFFFFFF          <1>       jmp   sysstime_3
16047                              <1>
16048                              <1> sysrename: ; Rename File (or Directory)
16049                              <1>       ; 19/01/2021
16050                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16051                              <1>       ;
16052                              <1>         ; INPUT ->
16053                              <1>         ;         EBX = File/Directory (ASCIIZ) name address
16054                              <1>       ;         ECX = New name (in same dir, no path name)
16055                              <1>       ; OUTPUT ->
16056                              <1>       ;          cf = 0 -> EAX = 0
16057                              <1>       ;          cf = 1 -> Error code in AL
16058                              <1>
16059                              <1>       ; 19/01/2021
16060 000128DF 803D[B0030300]00    <1>       cmp   byte [u.uid], 0  ; root (super user) ?
16061 000128E6 7614                <1>       jna   short sysrename_0
16062                              <1>
16063                              <1> sysrename_perm_err:
16064                              <1>       ;mov  dword [u.r0], ERR_PERM_DENIED
16065 000128E8 B80B000000          <1>       mov   eax, ERR_PERM_DENIED ; 'permission denied !'
16066 000128ED A3[64030300]        <1>       mov   [u.r0], eax
16067 000128F2 A3[C8030300]        <1>       mov   [u.error], eax
16068 000128F7 E9F0AFFFFF          <1>       jmp   error
16069                              <1>
16070                              <1> sysrename_0:
16071 000128FC 51                  <1>       push  ecx ; new file name address (in user space)
16072 000128FD 89DE                <1>       mov   esi, ebx
16073                              <1>       ; file name is forced, change directory as temporary
16074                              <1>       ;mov  ax, 1
16075                              <1>       ;mov  [FFF_Valid], ah ; 0 ; reset
16076                              <1>       ;call set_working_path
16077 000128FF E8BF060000          <1>       call  set_working_path_x
16078 00012904 731E                <1>       jnc   short sysrename_1
16079 00012906 21C0                <1>       and   eax, eax  ; 0 -> Bad Path!
16080 00012908 7505                <1>       jnz   short sysrename_err
16081                              <1>       ; eax = 0
16082                              <1> sysrename_path_not_found:
16083 0001290A B813000000          <1>       mov   eax, ERR_INV_PATH_NAME ; 'Bad path name !'
16084                              <1> sysrename_err:
16085 0001290F 59                  <1>       pop   ecx ; new file name address (in user space)
16086                              <1> sysrename_error:
```

```
16087 00012910 A3[64030300]        <1>       mov   [u.r0], eax
16088 00012915 A3[C8030300]        <1>       mov   [u.error], eax
16089 0001291A E879070000          <1>       call  reset_working_path
16090 0001291F E9C8AFFFFF          <1>       jmp   error
16091                              <1> sysrename_1:
16092 00012924 B008                <1>       mov   al, 08h ; Except volume labels (& long names)
16093 00012926 A0[94940100]        <1>       mov   al, [Attributes]
16094 0001292B 2410                <1>       and   al, 10h ;
16095                              <1>       ;mov  esi, FindFile_Name
16096                              <1>       ;mov  ax, 1800h ; Only files
16097                              <1>       ;mov  ax, 0810h ; Only directories
16098 0001292D 66B80008            <1>       mov   ax, 0800h ; Find File or Directory
16099 00012931 E85B6BFFFF          <1>       call  find_first_file
16100                              <1>       ;jnc  short sysrename_2
16101 00012936 72D7                <1>       jc    short sysrename_err
16102                              <1> sysrename_2:
16103                              <1>       ; ESI = Directory Entry (FindFile_DirEntry) Location
16104                              <1>       ; EDI = Directory Buffer Directory Entry Location
16105                              <1>       ; EAX = File Size
16106                              <1>       ;  BL = Attributes of The File/Directory
16107                              <1>       ;  BH = Long Name Yes/No Status (>0 is YES)
16108                              <1>       ;  DX > 0 : Ambiguous filename chars are used
16109                              <1>
16110 00012938 6621D2              <1>       and   dx, dx ; Ambiguous filename chars used sign (DX>0)
16111 0001293B 7407                <1>       jz    short sysrename_3
16112 0001293D B81A000000          <1>       mov   eax, ERR_INV_FILE_NAME ; 'invalid file name !'
16113 00012942 EBCB                <1>         jmp short sysrename_err
16114                              <1> sysrename_3:
16115                              <1>       ; EDI = Directory buffer entry offset/address
16116                              <1>       ; BL = File (or Directory) Attributes
16117                              <1>       ; mov  bl, [EDI+0Bh]
16118                              <1>
16119 00012944 5A                  <1>       pop   edx ; new file name address (in user space)
16120                              <1>
16121                              <1>       ; check file/directory attributes
16122 00012945 F6C307              <1>       test  bl, 7 ; system, hidden, readonly
16123 00012948 759E                <1>         jnz short sysrename_perm_err
16124                              <1> sysrename_4:
16125 0001294A 66817F0CA101        <1>       cmp   word [edi+DirEntry_NTRes], 01A1h ; Singlix FS
16126 00012950 7496                <1>       je    short sysrename_perm_err ; -temporary!-
16127                              <1>
16128                              <1>       ; save old file name & file info (FFF structure)
16129 00012952 BE[7E930100]        <1>       mov   esi, FindFile_Drv
16130 00012957 BF[C4940100]        <1>       mov   edi, SourceFile_Drv
16131 0001295C B920000000          <1>       mov   ecx, 128/4
16132 00012961 F3A5                <1>       rep   movsd
16133                              <1>
16134 00012963 89D6                <1>       mov   esi, edx ; new file name address (in user space)
16135 00012965 BF[44950100]        <1>       mov   edi, DestinationFile_Drv
16136 0001296A E8F38CFFFF          <1>       call  parse_path_name
16137 0001296F 729F                <1>       jc    short sysrename_error ; eax = 1 (Bad file name)
16138                              <1>
16139                              <1>       ; same drive ?
16140 00012971 A0[7E930100]        <1>       mov   al, [FindFile_Drv]
16141 00012976 3A05[44950100]      <1>       cmp   al, [DestinationFile_Drv]
16142                              <1>       ;jne  short sysrename_perm_err ; Permission denied
16143 0001297C 7509                <1>       jne   short sysrename_5 ; Bad file name
16144                              <1>
16145                              <1>       ; no path name !? (rename file in same directory)
16146 0001297E 803D[45950100]20    <1>       cmp   byte [DestinationFile_Directory], 20h
16147 00012985 7607                <1>       jna   short sysrename_6
16148                              <1> sysrename_5:
16149 00012987 B801000000          <1>       mov   eax, ERR_BAD_CMD_ARG ; 1 = Bad file name
16150                              <1>                                  ; (Bad argument)
16151 0001298C EB82                <1>       jmp   short sysrename_error
16152                              <1> sysrename_6:
16153 0001298E 803D[86950100]20    <1>       cmp   byte [DestinationFile_Name], 20h
16154 00012995 76F0                <1>       jna   short sysrename_5
16155                              <1>
16156 00012997 BE[86950100]        <1>       mov   esi, DestinationFile_Name
16157 0001299C E8AE6EFFFF          <1>       call  check_filename ; is it a valid msdos file name?
16158 000129A1 0F8269FFFFFF        <1>       jc    sysrename_error ; 26 = ERR_INV_FILE_NAME
16159                              <1>
16160                              <1>       ;mov  esi, DestinationFile_Name
16161 000129A7 66B80008            <1>       mov   ax, 0800h ; Find File or Directory
16162 000129AB E8E16AFFFF          <1>       call  find_first_file
16163 000129B0 720A                <1>       jc    short sysrename_7
16164                              <1>
16165 000129B2 B80E000000          <1>       mov   eax, ERR_FILE_EXISTS  ; file already exists !
16166 000129B7 E954FFFFFF          <1>       jmp   sysrename_error
16167                              <1> sysrename_7:
16168                              <1>       ; eax = 2 (File not found !)
16169 000129BC 3C02                <1>       cmp   al, 2 ; ERR_NOT_FOUND
16170 000129BE 0F854CFFFFFF        <1>       jne   sysrename_error
16171                              <1>
16172                              <1>       ; 31/12/2017
16173                              <1>       ; Following code is also part of 'rename_file' in
16174                              <1>       ; 'trdosk3.s' (MainProg's 'rename' command) ; 13/11/2017
16175 000129C4 BE[86950100]        <1>       mov   esi, DestinationFile_Name ; (Rename_NewName)
16176 000129C9 668B0D[3E950100]    <1>       mov   cx, [SourceFile_DirEntryNumber]
16177 000129D0 66A1[2A950100]      <1>       mov   ax, [SourceFile_DirEntry+20] ; First Cluster, HW
16178 000129D6 C1E010              <1>       shl   eax, 16
16179 000129D9 66A1[30950100]      <1>       mov   ax, [SourceFile_DirEntry+26] ; First Cluster, LW
16180 000129DF 0FB61D[13950100]    <1>       movzx ebx, byte [SourceFile_LongNameEntryLength]
16181 000129E6 E8BF94FFFF          <1>       call  rename_directory_entry
16182 000129EB 0F821FFFFFFF        <1>       jc    sysrename_error
16183                              <1>       ;xor  eax, eax
16184 000129F1 A3[64030300]        <1>       mov   [u.r0], eax ; 0
16185                              <1>       ;mov  [u.error], eax
16186 000129F6 E89D060000          <1>       call  reset_working_path
16187 000129FB E90CAFFFFF          <1>       jmp   sysret
16188                              <1>
16189                              <1> sysmem: ; Get Total&Free Memory amount
16190                              <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16191                              <1>       ;
```

```
16192                                  <1>         ; INPUT ->
16193                                  <1>       ;     none
16194                                  <1>       ; OUTPUT ->
16195                                  <1>       ;     EAX = Total memory count (in bytes)
16196                                  <1>       ;     EBX = Virtually available memory amount (in bytes)
16197                                  <1>       ;           = 4GB - CORE (4MB)
16198                                  <1>       ;     ECX = Free memory count (in bytes)
16199                                  <1>       ;     EDX = Calculated free memory count (in bytes)
16200                                  <1>
16201 00012A00 A1[1C8A0100]            <1>       mov   eax, [memory_size] ; in pages
16202 00012A05 C1E00C                  <1>       shl   eax, 12                ; in bytes
16203 00012A08 A3[64030300]            <1>       mov   [u.r0], eax
16204 00012A0D E84319FFFF              <1>       call  calc_free_mem
16205                                  <1>       ; edx = calculated free pages
16206                                  <1>       ; ecx = 0
16207 00012A12 8B2D[60030300]          <1>       mov   ebp, [u.usp]  ; EBP points to user's registers
16208 00012A18 C745100000C0FF          <1>       mov   dword [ebp+16], ECORE ; EBX (for user)
16209                                  <1>                           ; 0FFC00000h ; 4GB - 4MB
16210 00012A1F C1E20C                  <1>       shl   edx, 12
16211 00012A22 895514                  <1>       mov   [ebp+20], edx ; EDX (for user)
16212 00012A25 8B0D[208A0100]          <1>       mov   ecx, [free_pages]
16213 00012A2B C1E10C                  <1>       shl   ecx, 12      ; free bytes
16214 00012A2E 894D18                  <1>       mov   [ebp+24], ecx ; ECX (for user)
16215                                  <1>       ;mov  [free_pages], edx
16216 00012A31 E9D6AEFFFF              <1>       jmp   sysret
16217                                  <1>
16218                                  <1> sysprompt:
16219                                  <1>       ; Set TRDOS 386 Command Interpreter (MainProg) prompt
16220                                  <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16221                                  <1>       ;
16222                                  <1>         ; INPUT ->
16223                                  <1>       ;     EBX = 0 -> use default prompt
16224                                  <1>       ;     EBX > 0 -> prompt string (ASCIIZ) address
16225                                  <1>       ;             (Max. 11 characters except ZERO tail)
16226                                  <1>       ; OUTPUT ->
16227                                  <1>       ;     (EAX = 0)
16228                                  <1>       ;     CF = 0 -> Successful
16229                                  <1>       ;     CF = 1 -> Failed
16230                                  <1>
16231 00012A36 21DB                    <1>       and   ebx, ebx
16232 00012A38 750A                    <1>       jnz   short sysprompt_0
16233                                  <1>
16234 00012A3A E85664FFFF              <1>       call  default_command_prompt ; '['+'TRDOS'+']'
16235 00012A3F E9C8AEFFFF              <1>       jmp   sysret
16236                                  <1>
16237                                  <1> sysprompt_0:
16238 00012A44 31C0                    <1>       xor   eax, eax
16239 00012A46 A3[64030300]            <1>       mov   [u.r0], eax
16240 00012A4B 89DE                    <1>       mov   esi, ebx
16241 00012A4D B90C000000              <1>       mov   ecx, 12
16242 00012A52 89E5                    <1>       mov   ebp, esp
16243 00012A54 29CC                    <1>       sub   esp, ecx
16244 00012A56 49                      <1>       dec   ecx ; 11
16245 00012A57 89E7                    <1>       mov   edi, esp
16246 00012A59 E833F1FFFF              <1>       call  transfer_from_user_buffer
16247 00012A5E 7211                    <1>       jc    short sysprompt_err
16248 00012A60 803E20                  <1>       cmp   byte [esi], 20h
16249 00012A63 760C                    <1>       jna   short sysprompt_err
16250 00012A65 E83D64FFFF              <1>       call  set_command_prompt
16251 00012A6A 89EC                    <1>       mov   esp, ebp
16252 00012A6C E99BAEFFFF              <1>       jmp   sysret
16253                                  <1> sysprompt_err:
16254                                  <1> syspath_err:
16255 00012A71 89EC                    <1>       mov   esp, ebp
16256 00012A73 E974AEFFFF              <1>       jmp   error
16257                                  <1>
16258                                  <1> syspath:
16259                                  <1>       ; Get/Set Run Path
16260                                  <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16261                                  <1>       ;
16262                                  <1>         ; INPUT ->
16263                                  <1>       ;     EBX = 0 -> get path (to buffer address in ECX)
16264                                  <1>       ;     EBX > 0 -> set path
16265                                  <1>       ;         EBX = Path string buffer address (ASCIIZ)
16266                                  <1>       ;              (Path description except 'PATH=')
16267                                  <1>       ;     ECX = Buffer address (if EBX = 0)
16268                                  <1>       ;           (ECX will not be used if EBX > 0)
16269                                  <1>       ;     DL = Buffer size (0 = 256 byte)
16270                                  <1>       ;
16271                                  <1>       ; OUTPUT ->
16272                                  <1>       ;     CF = 0 -> Successful (EAX = String length)
16273                                  <1>       ;     CF = 1 -> Failed (EAX = 0)
16274                                  <1>       ;
16275                                  <1>       ; NOTE: 'PATH=' or 'PATH' must be excluded
16276                                  <1>       ;  (It must not be at the beginning of the string.)
16277                                  <1>
16278 00012A78 89E5                    <1>       mov   ebp, esp
16279 00012A7A 81EC00010000            <1>       sub   esp, 256
16280 00012A80 89E7                    <1>       mov   edi, esp
16281                                  <1>
16282 00012A82 31C0                    <1>       xor   eax, eax
16283 00012A84 A3[64030300]            <1>       mov   [u.r0], eax
16284                                  <1>
16285 00012A89 21DB                    <1>       and   ebx, ebx
16286 00012A8B 752E                    <1>       jnz   short syspath_0
16287                                  <1>
16288                                  <1>       ; EBX = 0 -> get run path
16289 00012A8D 89CB                    <1>       mov   ebx, ecx ; buffer addr (in user's mem space)
16290 00012A8F BE[72410100]            <1>       mov   esi, Cmd_Path  ; 'PATH' address
16291 00012A94 0FB6CA                  <1>       movzx ecx, dl
16292 00012A97 80E901                  <1>       sub   cl, 1 ; 0 -> 255, 1 -> 0
16293 00012A9A 6683D101                <1>       adc   cx, 1 ; 255 -> 256, 0 -> 1
16294                                  <1>       ; EDI = Output buffer
16295                                  <1>       ; CX = Buffer length
16296                                  <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
```

```
16297                                  <1>       ; ESI = 'PATH' address (with zero tail)
16298 00012A9E E8387CFFFF              <1>       call   get_environment_string
16299 00012AA3 72CC                    <1>       jc     short syspath_err
16300 00012AA5 89DF                    <1>       mov    edi, ebx ; User's buffer address
16301 00012AA7 89E6                    <1>       mov    esi, esp
16302                                  <1>       ; EDI = User's buffer address
16303                                  <1>       ; ECX = transfer (byte) count
16304 00012AA9 E899F0FFFF              <1>       call   transfer_to_user_buffer
16305 00012AAE 72C1                    <1>       jc     short syspath_err
16306 00012AB0 890D[64030300]          <1>       mov    [u.r0], ecx
16307 00012AB6 E951AEFFFF              <1>       jmp    sysret
16308                                  <1>
16309                                  <1> syspath_0:
16310 00012ABB 89DE                    <1>       mov    esi, ebx
16311 00012ABD 0FB6CA                  <1>       movzx  ecx, dl
16312 00012AC0 80E901                  <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
16313 00012AC3 6683D101                <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
16314 00012AC7 E8C5F0FFFF              <1>       call   transfer_from_user_buffer
16315 00012ACC 72A3                    <1>       jc     short syspath_err
16316                                  <1>       ;(*) 'PATH=' will be added to
16317                                  <1>       ;         the head of the string
16318 00012ACE 83EC08                  <1>       sub    esp, 8 ;(*)
16319 00012AD1 89FE                    <1>       mov    esi, edi ;(*)
16320 00012AD3 E8E77BFFFF              <1>       call   set_path_x ;(*)
16321 00012AD8 7297                    <1>       jc     short syspath_err
16322 00012ADA 8915[64030300]          <1>       mov    [u.r0], edx ; run path string length
16323 00012AE0 E927AEFFFF              <1>       jmp    sysret
16324                                  <1>
16325                                  <1> sysenv:
16326                                  <1>       ; Get/Set Environment Variables
16327                                  <1>       ; 31/12/2017 (TRDOS 386 = TRDOS v2.0)
16328                                  <1>       ;
16329                                  <1>         ; INPUT ->
16330                                  <1>       ;     EBX = 0 -> get (all) environment variables
16331                                  <1>       ;           (Required Buffer length = 512 bytes)
16332                                  <1>       ;     EBX > 0 -> set (one) environment variable
16333                                  <1>       ;           (If there is not a '=' after
16334                                  <1>       ;           the environment variable name, it will
16335                                  <1>       ;           accepted as 'get environment variable'.)
16336                                  <1>       ;            EBX = Buffer address
16337                                  <1>       ;     ECX = Buffer address (if EBX = 0)
16338                                  <1>       ;           (ECX will not be used if EBX > 0)
16339                                  <1>       ;           (Note: Buffer size is 512 bytes.)
16340                                  <1>       ;     DL = Buffer size (0 = 256 byte)
16341                                  <1>       ;          (For one envrionment variable)
16342                                  <1>       ;
16343                                  <1>       ; OUTPUT ->
16344                                  <1>       ;     (EAX = 0)
16345                                  <1>       ;     CF = 0 -> Successful (EAX = String length)
16346                                  <1>       ;     CF = 1 -> Failed (EAX = 0)
16347                                  <1>       ;
16348                                  <1>       ; Note: Environment variable name, for example,
16349                                  <1>       ;     'PATH=' must be included at the beginning
16350                                  <1>       ;     of the environment string. If the variable
16351                                  <1>       ;     name is as 'PATH' but it is not as 'PATH='
16352                                  <1>       ;     the variable string (row) will be returned.
16353                                  <1>       ;     If variable name is as 'PATH=' but there is
16354                                  <1>       ;     not a following text after the variable name,
16355                                  <1>       ;     the environment variable will be reset/deleted.
16356                                  <1>
16357 00012AE5 89E5                    <1>       mov    ebp, esp
16358 00012AE7 81EC00020000            <1>       sub    esp, 512
16359 00012AED 89E7                    <1>       mov    edi, esp
16360                                  <1>
16361 00012AEF 31C0                    <1>       xor    eax, eax
16362 00012AF1 A3[64030300]            <1>       mov    [u.r0], eax
16363                                  <1>
16364 00012AF6 21DB                    <1>       and    ebx, ebx
16365 00012AF8 7524                    <1>       jnz    short sysenv_0
16366                                  <1>
16367                                  <1>       ; EBX = 0 -> get (all) environment variables
16368 00012AFA 89EC                    <1>       mov    esp, ebp
16369 00012AFC BE00300900              <1>       mov    esi, Env_Page  ; Environment page
16370 00012B01 89CF                    <1>       mov    edi, ecx ; buffer addr (in user's mem space)
16371 00012B03 B900020000              <1>       mov    ecx, 512
16372 00012B08 E83AF0FFFF              <1>       call   transfer_to_user_buffer
16373 00012B0D 0F82D9ADFFFF            <1>       jc     error
16374 00012B13 890D[64030300]          <1>       mov    [u.r0], ecx
16375 00012B19 E9EEADFFFF              <1>       jmp    sysret
16376                                  <1>
16377                                  <1> sysenv_0:
16378 00012B1E 89DE                    <1>       mov    esi, ebx ; * ; user's buffer address
16379 00012B20 0FB6CA                  <1>       movzx  ecx, dl
16380 00012B23 80E901                  <1>       sub    cl, 1 ; 0 -> 255, 1 -> 0
16381 00012B26 6683D101                <1>       adc    cx, 1 ; 255 -> 256, 0 -> 1
16382 00012B2A E862F0FFFF              <1>       call   transfer_from_user_buffer
16383 00012B2F 723F                    <1>       jc     short sysenv_err
16384 00012B31 89FE                    <1>       mov    esi, edi
16385 00012B33 8A06                    <1>       mov    al, [esi]
16386 00012B35 3C20                    <1>       cmp    al, 20h
16387 00012B37 7637                    <1>       jna    short sysenv_err
16388 00012B39 3C3D                    <1>       cmp    al, '='
16389 00012B3B 7433                    <1>       je     short sysenv_err
16390 00012B3D 56                      <1>       push   esi
16391                                  <1> sysenv_1:
16392 00012B3E 46                      <1>       inc    esi
16393 00012B3F 803E3D                  <1>       cmp    byte [esi], '='
16394 00012B42 7433                    <1>       je     short sysenv_3
16395 00012B44 803E20                  <1>       cmp    byte [esi], 20h
16396 00012B47 73F5                    <1>       jnb    short sysenv_1
16397 00012B49 C60600                  <1>       mov    byte [esi], 0
16398 00012B4C 5E                      <1>       pop    esi
16399                                  <1>       ; EDI = Output buffer
16400                                  <1>       ; CX = Buffer length
16401 00012B4D 30C0                    <1>       xor    al, al
```

```
16402                                  <1>       ; AL = 0 -> use ASCIIZ word in [ESI]
16403                                  <1>       ; ESI = Environment variable name address
16404 00012B4F E8877BFFFF              <1>       call   get_environment_string
16405 00012B54 721A                    <1>       jc     short sysenv_err
16406 00012B56 89DF                    <1>       mov    edi, ebx ; * ; user's buffer address
16407 00012B58 89C1                    <1>       mov    ecx, eax ; String length
16408 00012B5A 89E6                    <1>       mov    esi, esp
16409                                  <1>       ; ESI = system buffer address
16410                                  <1>       ; EDI = User's buffer address
16411                                  <1>       ; ECX = transfer (byte) count
16412 00012B5C E8E6EFFFFF              <1>       call   transfer_to_user_buffer
16413 00012B61 720D                    <1>       jc     short sysenv_err
16414 00012B63 890D[64030300]          <1>       mov    [u.r0], ecx ; transfer (byte) count
16415                                  <1> sysenv_2:
16416 00012B69 89EC                    <1>       mov    esp, ebp
16417 00012B6B E99CADFFFF              <1>       jmp    sysret
16418                                  <1> sysenv_err:
16419 00012B70 89EC                    <1>       mov    esp, ebp
16420 00012B72 E975ADFFFF              <1>       jmp    error
16421                                  <1> sysenv_3:
16422 00012B77 46                      <1>       inc    esi
16423 00012B78 803E20                  <1>       cmp    byte [esi], 20h
16424 00012B7B 73FA                    <1>       jnb    short sysenv_3
16425 00012B7D C60600                  <1>       mov    byte [esi], 0
16426 00012B80 5E                      <1>       pop    esi
16427 00012B81 E8187CFFFF              <1>       call   set_environment_string
16428 00012B86 72E8                    <1>       jc     short sysenv_err
16429 00012B88 8915[64030300]          <1>       mov    [u.r0], edx
16430 00012B8E EBD9                    <1>       jmp    short sysenv_2
16431                                  <1>
16432                                  <1>
16433                                  <1> ; 22/01/2021
16434                                  <1> ; temporary - 24/01/2016
16435                                  <1>
16436                                  <1> iget:
16437                                  <1>       ;retn
16438                                  <1> isintr:
16439                                  <1>       ;retn
16440                                  <1> iopen:
16441                                  <1>       ;retn
16442                                  <1> iclose:
16443                                  <1>       ;retn
16444                                  <1> sndc:
16445                                  <1>       ;retn
16446                                  <1> access:
16447                                  <1>       ;retn
16448                                  <1> sleep:
16449 00012B90 C3                      <1>       retn
 3093                                      %include 'trdosk7.s' ; 24/01/2016
    1                                  <1> ; ****************************************************************************
    2                                  <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - DISK READ&WRITE : trdosk7.s
    3                                  <1> ; ----------------------------------------------------------------------------
    4                                  <1> ; Last Update: 25/02/2016
    5                                  <1> ; ----------------------------------------------------------------------------
    6                                  <1> ; Beginning: 24/01/2016
    7                                  <1> ; ----------------------------------------------------------------------------
    8                                  <1> ; Assembler: NASM version 2.11 (trdos386.s)
    9                                  <1> ; ----------------------------------------------------------------------------
   10                                  <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
   11                                  <1> ; DISK_IO.ASM (20/07/2011)
   12                                  <1> ; ****************************************************************************
   13                                  <1> ; DISK_IO.ASM (c) 2009-2011 Erdogan TAN [ 04/07/2009 ] Last Update: 20/07/2011
   14                                  <1>
   15                                  <1> disk_write:
   16                                  <1>       ; 25/02/2016
   17                                  <1>       ; 24/02/2016
   18                                  <1>       ; 23/02/2016
   19 00012B91 807E0500                <1>       cmp    byte [esi+LD_LBAYes], 0
   20 00012B95 777B                    <1>         ja      short lba_write
   21                                  <1>
   22                                  <1> chs_write:
   23                                  <1>       ; 25/02/2016
   24                                  <1>       ; 23/02/2016
   25 00012B97 C605[CD920100]03        <1>       mov    byte [disk_rw_op], 3 ; CHS write
   26 00012B9E EB0D                    <1>       jmp    short chs_rw
   27                                  <1>
   28                                  <1> disk_read:
   29                                  <1>       ; 25/02/2016
   30                                  <1>       ; 24/02/2016
   31                                  <1>       ; 23/02/2016
   32                                  <1>       ; 17/02/2016
   33                                  <1>       ; 14/02/2016
   34                                  <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
   35                                  <1>       ; 17/10/2010
   36                                  <1>       ; 18/04/2010
   37                                  <1>       ;
   38                                  <1>       ; INPUT -> EAX = Logical Block Address
   39                                  <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
   40                                  <1>       ;         ECX = Sector Count
   41                                  <1>       ;         EBX = Destination Buffer
   42                                  <1>       ; OUTPUT ->
   43                                  <1>       ;         cf = 0 or cf = 1
   44                                  <1>       ; (Modified registers: EAX, EBX, ECX, EDX)
   45                                  <1>
   46 00012BA0 807E0500                <1>       cmp    byte [esi+LD_LBAYes], 0
   47 00012BA4 7775                    <1>         ja      short lba_read
   48                                  <1>
   49                                  <1> chs_read:
   50                                  <1>       ; 25/02/2016
   51                                  <1>       ; 24/02/2016
   52                                  <1>       ; 23/02/2016
   53                                  <1>       ; 31/01/2016 (TRDOS 386 =  TRDOS v2.0)
   54                                  <1>       ; 20/07/2011
   55                                  <1>       ; 04/07/2009
   56                                  <1>       ;
```

```
                            <1>       ; INPUT -> EAX = Logical Block Address
                            <1>       ;         ECX = Number of sectors to read
                            <1>       ;         ESI = Logical Dos Disk Table Offset (DRV)
                            <1>       ;         EBX = Destination Buffer
                            <1>       ; OUTPUT ->
                            <1>       ;         cf = 0 or cf = 1
                            <1>       ; (Modified registers: EAX; EBX, ECX, EDX)
                            <1>
                            <1>       ; 23/02/2016
00012BA6 C605[CD920100]02   <1>       mov   byte [disk_rw_op], 2 ; CHS read
                            <1>
                            <1> chs_rw:
                            <1>       ;;movzx      edx, word [esi+LD_BPB+SecPerTrack]
                            <1>       ;movzx edx, byte [esi+LD_BPB+SecPerTrack] ; <= 63
                            <1>       ;mov   [disk_rw_spt], dl
                            <1>
                            <1> chs_read_next_sector:
00012BAD C605[CE920100]04   <1>       mov   byte [retry_count], 4
                            <1>
                            <1> chs_read_retry:
                            <1>       ;mov   [sector_count], ecx ; 23/02/2016
                            <1>
00012BB4 50                 <1>       push  eax                ; Linear sector #
00012BB5 51                 <1>       push  ecx                ; # of FAT/FILE/DIR sectors
                            <1>
00012BB6 0FB74E1E           <1>       movzx ecx, word [esi+LD_BPB+SecPerTrack]
                            <1>       ;movzx ecx, byte [disk_rw_spt] ; 23/02/2016
00012BBA 29D2               <1>       sub   edx, edx
00012BBC F7F1               <1>       div   ecx
                            <1>       ; eax = track, dx (dl ) = sector (on track)
                            <1>       ;sub   cl, dl ; 24/02/2016 (spt - sec)
                            <1>       ;push  ecx ; *
00012BBE 6689D1             <1>       mov   cx, dx             ; Sector (zero based)
00012BC1 6641               <1>       inc   cx                 ; To make it 1 based
00012BC3 6651               <1>       push  cx
00012BC5 668B4E20           <1>       mov   cx, [esi+LD_BPB+Heads]
00012BC9 6629D2             <1>       sub   dx, dx
00012BCC F7F1               <1>       div   ecx                ; Convert track to head & cyl
                            <1>       ; eax (ax) = cylinder, dx (dl) = head (max. FFh)
00012BCE 88D6               <1>       mov   dh, dl
00012BD0 6659               <1>       pop   cx                 ; AX=Cyl, DH=Head, CX=Sector
00012BD2 8A5602             <1>       mov   dl, [esi+LD_PhyDrvNo]
                            <1>
00012BD5 88C5               <1>       mov   ch, al             ; NOTE: max. 1023 cylinders !
00012BD7 C0CC02             <1>       ror   ah, 2              ; Rotate 2 bits right
00012BDA 08E1               <1>       or    cl, ah
                            <1>
                            <1>       ; 24/02/2016
                            <1>       ;pop   eax ; * (spt - sec) (example: 63 - 0 = 63)
                            <1>       ;cmp   eax, [sector_count]
                            <1>       ;jb    short chs_write_sectors
                            <1>       ;je    short chs_read_sectors
                            <1>       ;; (# of sectors to read is more than remaining sectors on the track)
                            <1>       ;mov   al, [sector_count]
                            <1> ;chs_read_sectors: ; read or write !
00012BDC B001               <1>       mov   al, 1 ; 25/02/2016
00012BDE 8A25[CD920100]     <1>       mov   ah, [disk_rw_op]  ; 02h = chs read, 03h = chs write
                            <1>       ;
00012BE4 E84826FFFF         <1>       call  int13h             ; BIOS Service func ( ah ) = 2
                            <1>                                ; Read disk sectors
                            <1>                                ; AL-sec num CH-track CL-sec
                            <1>                                ; DH-head DL-drive ES:BX-buffer
                            <1>                                ; CF-flag AH-stat AL-sec read
                            <1>                               ; If CF = 1 then (If AH > 0)
00012BE9 8825[CF920100]     <1>       mov   [disk_rw_err], ah
                            <1>
00012BEF 59                 <1>       pop   ecx
00012BF0 58                 <1>       pop   eax
00012BF1 7314               <1>       jnc   short chs_read_ok
                            <1>
00012BF3 803D[CF920100]09   <1>       cmp   byte [disk_rw_err], 09h ; DMA crossed 64K segment boundary
00012BFA 7408               <1>       je    short chs_read_error_retn
                            <1>
00012BFC FE0D[CE920100]     <1>       dec   byte [retry_count]
00012C02 75B0               <1>       jnz   short chs_read_retry
                            <1>
                            <1> chs_read_error_retn:
00012C04 F9                 <1>       stc
                            <1>       ;retn
00012C05 EB69               <1>       jmp   short update_drv_error_byte
                            <1>
                            <1> ;chs_write_sectors: ; read or write
                            <1>       ;; (# of sectors to read is less than remaining sectors on the track)
                            <1>       ;mov   [sector_count], al
                            <1>       ;jmp   short chs_read_sectors
                            <1>
                            <1> chs_read_ok:
                            <1>       ;; 23/02/2016
                            <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
00012C07 40                 <1>       inc   eax ; next sector
00012C08 81C300020000       <1>       add   ebx, 512
00012C0E E29D               <1>       loop  chs_read_next_sector
00012C10 EB5E               <1>       jmp   short update_drv_error_byte
                            <1>
                            <1> lba_write:
                            <1>       ; 23/02/2016
00012C12 C605[CD920100]1C   <1>       mov   byte [disk_rw_op], 1Ch ; LBA write
00012C19 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
00012C1B C605[CD920100]1B       <1>       mov   byte [disk_rw_op], 1Bh ; LBA read
                                <1>
                                <1> lba_rw:
                                <1>       ; 17/02/2016
00012C22 57                     <1>       push  edi
                                <1>
00012C23 890D[D0920100]         <1>       mov   [sector_count], ecx ; total sector (read) count
                                <1>
00012C29 8A5602                 <1>       mov   dl, [esi+LD_PhyDrvNo]
                                <1>       ; dl = physical drive number (0,1, 80h, 81h, 82h, 83h)
                                <1>
                                <1> lba_read_next:
00012C2C 81F900010000           <1>       cmp   ecx, 256
00012C32 7605                   <1>       jna   short lba_read_rsc
00012C34 B900010000             <1>       mov   ecx, 256 ; 17/02/2016
                                <1> lba_read_rsc:
00012C39 290D[D0920100]         <1>       sub   [sector_count], ecx ; remain sectors
                                <1>
00012C3F 89CF                   <1>       mov   edi, ecx
00012C41 89C1                   <1>       mov   ecx, eax ; sector number/address
                                <1>
00012C43 C605[CE920100]04       <1>       mov   byte [retry_count], 4
                                <1> lba_read_retry:
00012C4A 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
00012C4C 8A25[CD920100]         <1>       mov   ah, [disk_rw_op] ; 1Bh = LBA read, 1Ch = LBA write
00012C52 E8DA25FFFF             <1>       call  int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
00012C57 8825[CF920100]         <1>       mov   [disk_rw_err], ah
00012C5D 7334                   <1>       jnc   short lba_read_ok
00012C5F 80FC80                 <1>       cmp   ah, 80h ; time out?
00012C62 740A                   <1>         je      short lba_read_stc_retn
00012C64 FE0D[CE920100]         <1>       dec   byte [retry_count]
00012C6A 7FDE                   <1>       jg    short lba_read_retry
00012C6C 743A                   <1>       jz    short lba_read_reset
                                <1>       ; sf =  1
                                <1>
                                <1> lba_read_stc_retn:
00012C6E F9                     <1>       stc
                                <1> lba_read_retn:
00012C6F 5F                     <1>       pop   edi
                                <1>
                                <1> update_drv_error_byte:
00012C70 9C                     <1>       pushf
00012C71 53                     <1>       push  ebx
00012C72 6651                   <1>       push  cx
                                <1>       ;or   ecx, ecx
                                <1>       ;jz   short udrv_errb0
00012C74 8A0D[CF920100]         <1>       mov   cl, [disk_rw_err]
                                <1> udrv_errb0:
00012C7A 0FB65E02               <1>       movzx ebx, byte [esi+LD_PhyDrvNo]
00012C7E 80FB02                 <1>       cmp   bl, 2
00012C81 7203                   <1>         jb      short udrv_errb1
00012C83 80EB7E                 <1>       sub   bl, 7Eh
                                <1>       ;cmp  bl, 5
                                <1>       ;ja   short udrv_errb2
                                <1> udrv_errb1:
00012C86 81C3[616E0000]         <1>         add     ebx, drv.error ; 13/02/2016
00012C8C 880B                   <1>       mov   [ebx], cl ; error code
                                <1> udrv_errb2:
00012C8E 6659                   <1>       pop   cx
00012C90 5B                     <1>       pop   ebx
00012C91 9D                     <1>       popf
00012C92 C3                     <1>       retn
                                <1>
                                <1> lba_read_ok:
00012C93 89C8                   <1>       mov   eax, ecx ; sector number
00012C95 01F8                   <1>       add   eax, edi ; sector number (next)
00012C97 C1E709                 <1>       shl   edi, 9 ; sector count * 512
00012C9A 01FB                   <1>       add   ebx, edi ; next buffer offset
                                <1>
00012C9C 8B0D[D0920100]         <1>       mov   ecx, [sector_count] ; remaining sectors
00012CA2 09C9                   <1>       or    ecx, ecx
00012CA4 7586                   <1>       jnz   short lba_read_next
00012CA6 EBC7                   <1>       jmp   short lba_read_retn
```

```
                                <1>
                                <1> lba_read_reset:
00012CA8 B40D                   <1>       mov    ah, 0Dh ; Alternate reset
00012CAA E88225FFFF             <1>         call int13h
                                <1>       ; al = ? (changed)
                                <1>       ; ah = error code
00012CAF 7399                   <1>       jnc    short lba_read_retry
00012CB1 EBBC                   <1>       jmp    short lba_read_retn
                                    %include 'trdosk8.s' ; 24/01/2016
                                <1> ; ****************************************************************************
                                <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - MAIN PROGRAM : trdosk8.s
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Last Update: 12/02/2021
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Beginning: 24/01/2016
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Assembler: NASM version 2.11 (trdos386.s)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                                <1> ; u0.s (20/11/2015), u4.s (14/10/2015)
                                <1> ; ****************************************************************************
                                <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                                <1> ; TRDOS2.ASM (09/11/2011)
                                <1> ; ----------------------------------------------------------------------------
                                <1> ; DIR.ASM (c) 2004-2011 Erdogan TAN  [07/01/2004] Last Update: 09/10/2011
                                <1>
                                <1> set_run_sequence:
                                <1>       ; 23/12/2016
                                <1>       ; 10/06/2016
                                <1>       ; 22/05/2016
                                <1>       ; 20/05/2016
                                <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>       ; TRDOS 386 feature only !
                                <1>       ;
                                <1>       ; INPUT ->
                                <1>       ;     AL = process number (next process)
                                <1>       ;
                                <1>       ;     this process must be added to run sequence
                                <1>       ;
                                <1>       ;     [u.pri] = priority of present process
                                <1>       ;
                                <1>       ;     DL = priority (queue)
                                <1>       ;          0 = background (low) ; run on background
                                <1>       ;          1 = regular (normal) ; run as regular
                                <1>       ;          2 = event (high)     ; run for event
                                <1>       ;
                                <1>       ;     1) If the requested process is already running:
                                <1>       ;        a) If present priority is high ([u.pri]=2)
                                <1>       ;           and requested priority is also high,
                                <1>       ;           there is nothing to do! Because it has been
                                <1>       ;           done already (before this attempt).
                                <1>       ;        b) If present priority is high ([u.pri]=2)
                                <1>       ;           and requested priority is not high, there is
                                <1>       ;           nothing to do! Because, it's current
                                <1>       ;           run queue is unspecified, here. (It may be in
                                <1>       ;           a waiting list or in a run queue; if the new
                                <1>       ;           priority would be used to add it to relavant
                                <1>         ;             run queue, this would be wrong, unnecessary
                                <1>       ;           and destabilizing duplication!)
                                <1>       ;        c) If present priority is not high ([u.pri]<2)
                                <1>         ;             and requested priority is high (event),
                                <1>       ;           process will be added to present priority's
                                <1>       ;           run queue and then, priority will be changed
                                <1>       ;           to high ([u.pri]=2).
                                <1>       ;        d) If present priority is not high ([u.pri]<2)
                                <1>       ;           and requested priority is not high, [u.pri]
                                <1>       ;           value will be changed. There is nothing to do
                                <1>       ;           in addition. (The new priority value will be
                                <1>       ;           used by 'tswap/tswitch' procedure at 'sysret'
                                <1>       ;           or 'sysrele' stage.)
                                <1>       ;
                                <1>       ;     2) If the requested process is not running:
                                <1>       ;        a) If requested priority of the requested
                                <1>       ;           (next) process is high (event) and priority
                                <1>       ;           of present process is not high, the requested
                                <1>       ;           process will be added to ('runq_event') high
                                <1>       ;           priority run queue and then present (running)
                                <1>       ;           process will be stopped (swapped/switched out)
                                <1>       ;           immediately if it is in user mode, or it's
                                <1>       ;           [u.quant] value will be reset to 0 and (then)
                                <1>       ;           it will be stopped at 'sysret' stage.
                                <1>       ;        b) If requested priority of the requested
                                <1>       ;           (next) process is high (event) and priority
                                <1>       ;           of present process is also high, the requested
                                <1>       ;           process will be added to ('runq_event') high
                                <1>       ;           priority run queue and present (running)
                                <1>       ;           process will be allowed to run until it's
                                <1>       ;           time quantum will be elapsed ([u.quant]=0).
                                <1>       ;        c) If requested priority of the requested
                                <1>       ;           (next) process is not high ('run for event'),
                                <1>       ;           there is nothing to do. Because, it's current
                                <1>       ;           run queue is unspecified, here. (It may be in
                                <1>       ;           a waiting list or in a run queue; if the new
                                <1>       ;           priority would be used to add it to relavant
                                <1>         ;             run queue, this would be wrong, unnecessary
                                <1>       ;           and destabilizing duplication!)
                                <1>       ;
                                <1>       ; OUTPUT ->
                                <1>       ;     none
                                <1>       ;
                                <1>       ;     [u.pri] = priority of present process
                                <1>       ;
                                <1>       ;     cf = 1, if the request could not be fulfilled.
                                <1>       ;
                                <1>       ;     NOTE:
```

```
                                        <1>         ;          * Processes in 'run as regular' queue can run
                                        <1>       ;            if there is no process in 'run for event' queue
                                        <1>       ;            ('run for event' processes have higher priority)
                                        <1>       ;          * When [u.quant] time quantum of a process is
                                        <1>       ;            elapsed, it's high priority ('run for event')
                                        <1>       ;            status will be disabled, it can be run in sequence
                                        <1>       ;            of it's actual run queue.
                                        <1>       ;          * A 'run on background' process will always be
                                        <1>       ;            sequenced in 'run on background' (low priority)
                                        <1>       ;            queue, it can run only when other priority queues
                                        <1>       ;            are empty. (idle time processes, e.g. printing)
                                        <1>       ;
                                        <1>       ; Modified registers: eax, ebx, edx
                                        <1>       ;
                                        <1>
                                        <1> srunseq_0:
00012CB3 3A05[B3030300]                 <1>         cmp     al, [u.uno]     ; same process ?
00012CB9 750C                           <1>       jne    short srunseq_2 ; no
                                        <1>
00012CBB 8A25[A9030300]                 <1>       mov    ah, [u.pri]  ; present/current priority
00012CC1 80FC02                         <1>       cmp    ah, 2        ; 'run for event' priority level
00012CC4 7221                           <1>       jb     short srunseq_6 ; no
                                        <1>
                                        <1> srunseq_1:
                                        <1>       ; there is nothing to do!
00012CC6 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
00012CC7 80FA02                         <1>       cmp    dl, 2        ; event queue
00012CCA 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
00012CCC 3A15[A9030300]                 <1>       cmp    dl, [u.pri]  ; priority of present process
00012CD2 7606                           <1>       jna    short srunseq_3 ; is high, also
                                        <1>       ;
                                        <1>       ; present process will be swapped/switched out
00012CD4 FE05[A9960100]                 <1>       inc    byte [p_change] ; 1
                                        <1>
                                        <1> srunseq_3:
                                        <1>       ; add process to 'runq_event' queue for new event
00012CDA BB[52030300]                   <1>       mov    ebx, runq_event ; high priority run queue
                                        <1>
                                        <1> srunseq_4:
                                        <1>       ; al = process number
                                        <1>       ; ebx = run queue
00012CDF E887EDFFFF                     <1>       call   putlu
00012CE4 C3                             <1>       retn
                                        <1>
                                        <1> srunseq_5:
00012CE5 F5                             <1>       cmc
00012CE6 C3                             <1>       retn
                                        <1>
                                        <1> srunseq_6:
                                        <1>       ; present priority of the process is not high
                                        <1>
00012CE7 8815[A9030300]                 <1>       mov    [u.pri], dl ; new priority
                                        <1>                          ; (will be used by 'tswap')
                                        <1>
00012CED 80FA02                         <1>       cmp    dl, 2        ; high priority ?
00012CF0 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>
00012CF2 BB[54030300]                   <1>       mov    ebx, runq_normal ; 'run as regular' queue
                                        <1>
00012CF7 20E4                           <1>       and    ah, ah  ;  previous value of [u.pri]
00012CF9 75E4                           <1>       jnz    short srunseq_4
                                        <1>
00012CFB 43                             <1>       inc    ebx
00012CFC 43                             <1>       inc    ebx
                                        <1>       ; ebx = runq_background ; 'run on backgroud' queue
                                        <1>
00012CFD 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>
00012CFF 803D[A8030300]00               <1>       cmp    byte [u.quant], 0
00012D06 772C                           <1>       ja     short clk_1
                                        <1>       ;
00012D08 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.
00012D0F 7623                           <1>       jna    short clk_1 ; yes, do not swap out
                                        <1>       ;
00012D11 803D[5B030300]FF               <1>       cmp     byte [sysflg], 0FFh ; user or system space ?
00012D18 7520                           <1>       jne    short clk_2      ; system space (sysflg <> 0FFh)
                                        <1>       ;
00012D1A 66833D[AA030300]00             <1>       cmp    word [u.intr], 0
```

```
00012D22 7616                <1>       jna    short clk_2
                             <1>       ;
                             <1>       ; 23/05/2016
00012D24 803D[AA960100]00    <1>       cmp    byte [multi_tasking], 0
00012D2B 760D                <1>       jna    short clk_2
                             <1>       ;
00012D2D FE05[A9960100]      <1>       inc    byte [p_change] ; it is time to change running process
00012D33 C3                  <1>       retn
                             <1> clk_1:
00012D34 FE0D[A8030300]      <1>       dec    byte [u.quant]
                             <1> clk_2:
00012D3A 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
00012D3B 80FC07              <1>       cmp    ah, 7
00012D3E 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
00012D40 FB                  <1>       sti
                             <1>
                             <1>       ;sti   ; enable interrupts
00012D41 80642408FE          <1>       and    byte [esp+8], 11111110b   ; clear carry bit of eflags register
                             <1>
00012D46 80FC01              <1>       cmp    ah, 1
00012D49 7205                <1>       jb     short int34h_0
00012D4B 7705                <1>       ja     short int34h_1
                             <1>
00012D4D EE                  <1>       out    dx, al
                             <1>       ;iretd
00012D4E EB01                <1>       jmp short int34h_iret
                             <1>
                             <1> int34h_0:
00012D50 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
00012D51 CF                  <1>       iretd
                             <1>
                             <1> int34h_1:
00012D52 F6C401              <1>       test   ah, 1
00012D55 7516                <1>       jnz    short int34h_3 ; out
                             <1>
                             <1>       ; in
00012D57 80FC02              <1>       cmp    ah, 2
00012D5A 7707                <1>       ja     short int34h_2
                             <1>
00012D5C 6689D8              <1>       mov    ax, bx
00012D5F 66ED                <1>       in     ax, dx
                             <1>       ;iretd
00012D61 EBEE                <1>       jmp    short int34h_iret
                             <1>
                             <1> int34h_2:
00012D63 80FC04              <1>       cmp    ah, 4
00012D66 772C                <1>       ja     short int34h_7      ; 12/10/2017
                             <1>       ; ah = 4
00012D68 89D8                <1>       mov    eax, ebx
00012D6A ED                  <1>       in     eax, dx
```

```
                        <1>       ;iretd
00012D6B EBE4           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_3:
00012D6D 80FC03         <1>       cmp    ah, 3
00012D70 7707           <1>       ja     short int34h_4
                        <1>
00012D72 6689D8         <1>       mov    ax, bx
00012D75 66EF           <1>       out    dx, ax
                        <1>       ;iretd
00012D77 EBD8           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_4:
00012D79 80FC05         <1>       cmp    ah, 5
00012D7C 770B           <1>       ja     short int34h_6      ; 12/10/2017
                        <1>       ; ah = 5
00012D7E 89D8           <1>       mov    eax, ebx
00012D80 EF             <1>       out    dx, eax
                        <1>       ;iretd
00012D81 EBCE           <1>       jmp    short int34h_iret
                        <1>
                        <1> int34h_5:
00012D83 804C240801     <1>       or     byte [esp+8], 1     ; set carry bit of eflags register
00012D88 CF             <1>       iretd
                        <1>
                        <1>       ; 12/10/2017
                        <1> int34h_6:
00012D89 6689D8         <1>       mov    ax, bx
00012D8C EE             <1>       out    dx, al
00012D8D EB00           <1>       jmp    short $+2
00012D8F 86E0           <1>       xchg   ah, al
00012D91 EE             <1>       out    dx, al
                        <1>       ;xchg  al, ah
                        <1>       ;iretd
00012D92 EB06           <1>       jmp    short int34h_8
                        <1> int34h_7:
00012D94 EC             <1>       in     al, dx
00012D95 EB00           <1>       jmp    short $+2
00012D97 88C4           <1>       mov    ah, al
00012D99 EC             <1>       in     al, dx
                        <1> int34h_8:
00012D9A 86C4           <1>       xchg   al, ah
00012D9C CF             <1>       iretd
                        <1>
                        <1>
                        <1> INT4Ah:
                        <1>       ; 24/01/2016
                        <1>       ; this procedure will be called by 'RTC_INT' (in 'timer.s')
00012D9D C3             <1>       retn
                        <1>
                        <1> ; u0.s
                        <1> ; Retro UNIX 386 v1 Kernel (v0.2) - SYS0.INC
                        <1> ; Last Modification: 20/11/2015
                        <1>
                        <1> com2_int:
                        <1>       ; 07/11/2015
                        <1>       ; 24/10/2015
                        <1>       ; 23/10/2015
                        <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
                        <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
                        <1>       ; < serial port 2 interrupt handler >
                        <1>       ;
00012D9E 890424         <1>       mov    [esp], eax ; overwrite call return address
                        <1>       ;push  eax
00012DA1 66B80900       <1>       mov    ax, 9
00012DA5 EB07           <1>       jmp    short comm_int
                        <1> com1_int:
                        <1>       ; 07/11/2015
                        <1>       ; 24/10/2015
00012DA7 890424         <1>       mov    [esp], eax ; overwrite call return address
                        <1>       ; 23/10/2015
                        <1>       ;push  eax
00012DAA 66B80800       <1>       mov    ax, 8
                        <1> comm_int:
                        <1>       ; 20/11/2015
                        <1>       ; 18/11/2015
                        <1>       ; 17/11/2015
                        <1>       ; 16/11/2015
                        <1>       ; 09/11/2015
                        <1>       ; 08/11/2015
                        <1>       ; 07/11/2015
                        <1>       ; 06/11/2015 (serial4.asm, 'serial')
                        <1>       ; 01/11/2015
                        <1>       ; 26/10/2015
                        <1>       ; 23/10/2015
00012DAE 53             <1>       push   ebx
00012DAF 56             <1>       push   esi
00012DB0 57             <1>       push   edi
00012DB1 1E             <1>       push   ds
00012DB2 06             <1>       push   es
                        <1>       ; 18/11/2015
00012DB3 0F20DB         <1>       mov    ebx, cr3
00012DB6 53             <1>       push   ebx ; ****
                        <1>       ;
00012DB7 51             <1>       push   ecx ; ***
00012DB8 52             <1>       push   edx ; **
                        <1>       ;
00012DB9 BB10000000     <1>       mov    ebx, KDATA
00012DBE 8EDB           <1>       mov    ds, bx
00012DC0 8EC3           <1>       mov    es, bx
                        <1>       ;
00012DC2 8B0D[188A0100] <1>       mov    ecx, [k_page_dir]
00012DC8 0F22D9         <1>       mov    cr3, ecx
                        <1>       ; 20/11/2015
                        <1>       ; Interrupt identification register
```

```
00012DCB 66BAFA02            <1>       mov    dx, 2FAh ; COM2
                             <1>       ;
00012DCF 3C08                <1>       cmp    al, 8
00012DD1 7702                <1>       ja     short com_i0
                             <1>       ;
                             <1>       ; 20/11/2015
                             <1>       ; 17/11/2015
                             <1>       ; 16/11/2015
                             <1>       ; 15/11/2015
                             <1>       ; 24/10/2015
                             <1>       ; 14/03/2015 (Retro UNIX 386 v1 - Beginning)
                             <1>       ; 28/07/2014 (Retro UNIX 8086 v1)
                             <1>       ; < serial port 1 interrupt handler >
                             <1>       ;
00012DD3 FEC6                <1>       inc    dh ; 3FAh ; COM1 Interrupt id. register
                             <1> com_i0:
                             <1>       ;push  eax ; *
                             <1>       ; 07/11/2015
00012DD5 A2[828A0100]        <1>       mov    byte [ccomport], al
                             <1>       ; 09/11/2015
00012DDA 0FB7D8              <1>       movzx  ebx, ax ; 8 or 9
                             <1>       ; 17/11/2015
                             <1>       ; reset request for response status
00012DDD 88A3[788A0100]      <1>       mov    [ebx+req_resp-8], ah ; 0
                             <1>       ;
                             <1>       ; 20/11/2015
00012DE3 EC                  <1>       in     al, dx       ; read interrupt id. register
00012DE4 EB00                <1>       JMP    $+2          ; I/O DELAY
00012DE6 2404                <1>       and    al, 4        ; received data available?
00012DE8 7470                <1>       jz     short com_eoi; (transmit. holding reg. empty)
                             <1>       ;
                             <1>       ; 20/11/2015
00012DEA 80EA02              <1>       sub    dl, 3FAh-3F8h; data register (3F8h, 2F8h)
00012DED EC                  <1>       in     al, dx       ; read character
                             <1>       ;JMP   $+2          ; I/O DELAY
                             <1>       ; 08/11/2015
                             <1>       ; 07/11/2015
00012DEE 89DE                <1>       mov    esi, ebx
00012DF0 89DF                <1>       mov    edi, ebx
00012DF2 81C6[7C8A0100]      <1>       add    esi, rchar - 8 ; points to last received char
00012DF8 81C7[7E8A0100]      <1>       add    edi, schar - 8 ; points to last sent char
00012DFE 8806                <1>       mov    [esi], al ; received char (current char)
                             <1>       ; query
00012E00 20C0                <1>       and    al, al
00012E02 7527                <1>       jnz    short com_i2
                             <1>       ; response
                             <1>       ; 17/11/2015
                             <1>       ; set request for response status
00012E04 FE83[788A0100]      <1>         inc     byte [ebx+req_resp-8] ; 1
                             <1>       ;
00012E0A 6683C205            <1>       add    dx, 3FDh-3F8h; (3FDh, 2FDh)
00012E0E EC                  <1>       in     al, dx       ; read line status register
00012E0F EB00                <1>       JMP    $+2          ; I/O DELAY
00012E11 2420                <1>       and    al, 20h             ; transmitter holding reg. empty?
00012E13 7445                <1>       jz     short com_eoi       ; no
00012E15 B0FF                <1>       mov    al, 0FFh     ; response
00012E17 6683EA05            <1>       sub    dx, 3FDh-3F8h       ; data port (3F8h, 2F8h)
00012E1B EE                  <1>       out    dx, al       ; send on serial port
                             <1>       ; 17/11/2015
00012E1C 803F00              <1>       cmp    byte [edi], 0   ; query ? (schar)
00012E1F 7502                <1>       jne    short com_i1    ; no
00012E21 8807                <1>       mov    [edi], al    ; 0FFh (responded)
                             <1> com_i1:
                             <1>       ; 17/11/2015
                             <1>       ; reset request for response status (again)
00012E23 FE8B[788A0100]      <1>         dec     byte [ebx+req_resp-8] ; 0
00012E29 EB2F                <1>       jmp    short com_eoi
                             <1> com_i2:
                             <1>       ; 08/11/2015
00012E2B 3CFF                <1>       cmp    al, 0FFh     ; (response ?)
00012E2D 7417                <1>       je     short com_i3 ; (check for response signal)
                             <1>       ; 07/11/2015
00012E2F 3C04                <1>       cmp    al, 04h      ; EOT
00012E31 751C                <1>       jne    short com_i4
                             <1>       ; EOT = 04h (End of Transmit) - 'CTRL + D'
                             <1>       ;(an EOT char is supposed as a ctrl+brk from the terminal)
                             <1>       ; 08/11/2015
                             <1>             ; ptty -> tty 0 to 7 (pseudo screens)
00012E33 861D[468A0100]      <1>       xchg   bl, [ptty]  ; tty number (8 or 9)
00012E39 E8B247FFFF          <1>       call   ctrlbrk
00012E3E 861D[468A0100]      <1>       xchg   [ptty], bl ; (restore ptty value and BL value)
                             <1>       ;mov   al, 04h ; EOT
                             <1>       ; 08/11/2015
00012E44 EB09                <1>       jmp    short com_i4
                             <1> com_i3:
                             <1>       ; 08/11/2015
                             <1>       ; If 0FFh has been received just after a query
                             <1>       ; (schar, ZERO), it is a response signal.
                             <1>       ; 17/11/2015
00012E46 803F00              <1>         cmp     byte [edi], 0 ; query ? (schar)
00012E49 7704                <1>       ja     short com_i4 ; no
                             <1>       ; reset query status (schar)
00012E4B 8807                <1>       mov    [edi], al ; 0FFh
00012E4D FEC0                <1>       inc    al ; 0
                             <1> com_i4:
                             <1>       ; 27/07/2014
                             <1>       ; 09/07/2014
00012E4F D0E3                <1>       shl    bl, 1
00012E51 81C3[488A0100]      <1>       add    ebx, ttychr
                             <1>       ; 23/07/2014 (always overwrite)
                             <1>       ;;cmp  word [ebx], 0
                             <1>       ;;ja   short com_eoi
                             <1>       ;
00012E57 668903              <1>       mov    [ebx], ax   ; Save ascii code
                             <1>                          ; scan code = 0
```

```
                        <1> com_eoi:
                        <1>       ;mov   al, 20h
                        <1>       ;out   20h, al         ; end of interrupt
                        <1>       ;
                        <1>       ; 07/11/2015
                        <1>             ;pop   eax ; *
00012E5A A0[828A0100]   <1>       mov    al, byte [ccomport] ; current COM port
                        <1>        ; al = tty number (8 or 9)
00012E5F E85E010000     <1>         call wakeup
                        <1> com_iret:
                        <1>       ; 23/10/2015
00012E64 5A             <1>       pop    edx ; **
00012E65 59             <1>       pop    ecx ; ***
                        <1>       ; 18/11/2015
                        <1>       ;pop   eax ; ****
                        <1>       ;mov   cr3, eax
                        <1>       ;jmp   iiret
00012E66 E948DFFEFF     <1>       jmp    iiretp
                        <1>
                        <1> ;iiretp: ; 01/09/2015
                        <1> ;     ; 28/08/2015
                        <1> ;     pop    eax ; (*) page directory
                        <1> ;     mov    cr3, eax
                        <1> ;iiret:
                        <1> ;     ; 22/08/2014
                        <1> ;     mov    al, 20h ; END OF INTERRUPT COMMAND TO 8259
                        <1> ;     out    20h, al      ; 8259 PORT
                        <1> ;     ;
                        <1> ;     pop    es
                        <1> ;     pop    ds
                        <1> ;     pop    edi
                        <1> ;     pop    esi
                        <1> ;     pop    ebx ; 29/08/2014
                        <1> ;     pop    eax
                        <1> ;     iretd
                        <1>
                        <1> sp_init:
                        <1>       ; 07/11/2015
                        <1>       ; 29/10/2015
                        <1>       ; 26/10/2015
                        <1>       ; 23/10/2015
                        <1>       ; 29/06/2015
                        <1>       ; 14/03/2015 (Retro UNIX 386 v1 - 115200 baud)
                        <1>       ; 28/07/2014 (Retro UNIX 8086 v1 - 9600 baud)
                        <1>       ; Initialization of Serial Port Communication Parameters
                        <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
                        <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
                        <1>       ;
                        <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
                        <1>       ;
                        <1>       ; INPUT:  (29/06/2015)
                        <1>       ;     AL = 0 for COM1
                        <1>       ;          1 for COM2
                        <1>       ;     AH = Communication parameters
                        <1>       ;
                        <1>       ;  (*) Communication parameters (except BAUD RATE):
                        <1>       ;     Bit    4     3     2     1     0
                        <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
                        <1>       ;  this one -->    00 = none    0 = 1 bit  11 = 8 bits
                        <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
                        <1>       ;            11 = even
                        <1>       ;  Baud rate setting bits: (29/06/2015)
                        <1>       ;            Retro UNIX 386 v1 feature only !
                        <1>       ;     Bit    7    6    5  | Baud rate
                        <1>       ;            ------------------------
                        <1>       ;     value  0    0    0  | Default (Divisor = 1)
                        <1>       ;            0    0    1  | 9600 (12)
                        <1>       ;            0    1    0  | 19200 (6)
                        <1>       ;            0    1   1  | 38400 (3)
                        <1>       ;            1    0   0  | 14400 (8)
                        <1>       ;            1    0   1  | 28800 (4)
                        <1>       ;            1    1    0  | 57600 (2)
                        <1>       ;            1    1    1  | 115200 (1)
                        <1>
                        <1>       ; References:
                        <1>       ; (1) IBM PC-XT Model 286 BIOS Source Code
                        <1>       ;     RS232.ASM --- 10/06/1985 COMMUNICATIONS BIOS (RS232)
                        <1>       ; (2) Award BIOS 1999 - ATORGS.ASM
                        <1>       ; (3) http://wiki.osdev.org/Serial_Ports
                        <1>       ;
                        <1>       ; Set communication parameters for COM1 (= 03h)
                        <1>       ;
00012E6B BB[7E8A0100]   <1>       mov    ebx, com1p          ; COM1 parameters
00012E70 66BAF803       <1>       mov    dx, 3F8h            ; COM1
                        <1>        ; 29/10/2015
00012E74 66B90103       <1>       mov    cx, 301h  ; divisor = 1 (115200 baud)
00012E78 E86F000000     <1>       call   sp_i3  ; call A4
00012E7D A880           <1>       test   al, 80h
00012E7F 7410           <1>       jz     short sp_i0 ; OK..
                        <1>              ; Error !
                        <1>       ;mov   dx, 3F8h
00012E81 80EA05         <1>       sub    dl, 5 ; 3FDh -> 3F8h
00012E84 66B90E03       <1>       mov    cx, 30Eh  ; divisor = 12 (9600 baud)
00012E88 E85F000000     <1>       call   sp_i3  ; call A4
00012E8D A880           <1>       test   al, 80h
00012E8F 7508           <1>       jnz    short sp_i1
                        <1> sp_i0:
                        <1>         ; (Note: Serial port interrupts will be disabled here...)
                        <1>         ; (INT 14h initialization code disables interrupts.)
                        <1>       ;
00012E91 C603E3         <1>       mov    byte [ebx], 0E3h ; 11100011b
00012E94 E8DC000000     <1>       call   sp_i5 ; 29/06/2015
                        <1> sp_i1:
00012E99 43             <1>       inc    ebx
00012E9A 66BAF802       <1>       mov    dx, 2F8h            ; COM2
```

```
                        <1>         ; 29/10/2015
00012E9E 66B90103       <1>         mov     cx, 301h  ; divisor = 1 (115200 baud)
00012EA2 E845000000     <1>         call    sp_i3  ; call A4
00012EA7 A880           <1>         test    al, 80h
00012EA9 7410           <1>         jz      short sp_i2 ; OK..
                        <1>                 ; Error !
                        <1>         ;mov    dx, 2F8h
00012EAB 80EA05         <1>         sub     dl, 5 ; 2FDh -> 2F8h
00012EAE 66B90E03       <1>         mov     cx, 30Eh  ; divisor = 12 (9600 baud)
00012EB2 E835000000     <1>         call    sp_i3  ; call A4
00012EB7 A880           <1>         test    al, 80h
00012EB9 7530           <1>         jnz     short sp_i7
                        <1> sp_i2:
00012EBB C603E3         <1>         mov     byte [ebx], 0E3h ; 11100011b
                        <1> sp_i6:
                        <1>         ;; COM2 - enabling IRQ 3
                        <1>         ; 07/11/2015
                        <1>         ; 26/10/2015
00012EBE 9C             <1>         pushf
00012EBF FA             <1>         cli
                        <1>         ;
00012EC0 66BAFC02       <1>         mov     dx, 2FCh            ; modem control register
00012EC4 EC             <1>         in      al, dx                    ; read register
00012EC5 EB00           <1>         JMP     $+2                 ; I/O DELAY
00012EC7 0C08           <1>         or      al, 8               ; enable bit 3 (OUT2)
00012EC9 EE             <1>         out     dx, al              ; write back to register
00012ECA EB00           <1>         JMP     $+2                 ; I/O DELAY
00012ECC 66BAF902       <1>         mov     dx, 2F9h            ; interrupt enable register
00012ED0 EC             <1>         in      al, dx              ; read register
00012ED1 EB00           <1>         JMP     $+2                 ; I/O DELAY
                        <1>         ;or     al, 1               ; receiver data interrupt enable and
00012ED3 0C03           <1>         or      al, 3               ; transmitter empty interrupt enable
00012ED5 EE             <1>         out     dx, al                    ; write back to register
00012ED6 EB00           <1>         JMP     $+2                 ; I/O DELAY
00012ED8 E421           <1>         in      al, 21h             ; read interrupt mask register
00012EDA EB00           <1>         JMP     $+2                 ; I/O DELAY
00012EDC 24F7           <1>         and     al, 0F7h            ; enable IRQ 3 (COM2)
00012EDE E621           <1>         out     21h, al             ; write back to register
                        <1>         ;
                        <1>         ; 23/10/2015
00012EE0 B8[9E2D0100]   <1>         mov     eax, com2_int
00012EE5 A3[BD2F0100]   <1>         mov     [com2_irq3], eax
                        <1>         ; 26/10/2015
00012EEA 9D             <1>         popf
                        <1> sp_i7:
00012EEB C3             <1>         retn
                        <1>
                        <1> sp_i3:
                        <1> ;A4:            ;----- INITIALIZE THE COMMUNICATIONS PORT
                        <1>         ; 28/10/2015
00012EEC FEC2           <1>         inc     dl     ; 3F9h (2F9h); 3F9h, COM1 Interrupt enable register
00012EEE B000           <1>         mov     al, 0
00012EF0 EE             <1>         out     dx, al              ; disable serial port interrupt
00012EF1 EB00           <1>         JMP     $+2                 ; I/O DELAY
00012EF3 80C202         <1>         add     dl, 2  ; 3FBh (2FBh); COM1 Line control register (3FBh)
00012EF6 B080           <1>         mov     al, 80h
00012EF8 EE             <1>         out     dx, al              ; SET DLAB=1 ; divisor latch access bit
                        <1>         ;----- SET BAUD RATE DIVISOR
                        <1>         ; 26/10/2015
00012EF9 80EA03         <1>         sub     dl, 3   ; 3F8h (2F8h)     ; register for least significant byte
                        <1>                                     ; of the divisor value
00012EFC 88C8           <1>         mov     al, cl ; 1
00012EFE EE             <1>         out     dx, al              ; 1 = 115200 baud (Retro UNIX 386 v1)
                        <1>                                     ; 2 = 57600 baud
                        <1>                                     ; 3 = 38400 baud
                        <1>                                     ; 6 = 19200 baud
                        <1>                                     ; 12 = 9600 baud (Retro UNIX 8086 v1)
00012EFF EB00           <1>         JMP     $+2                 ; I/O DELAY
00012F01 28C0           <1>         sub     al, al
00012F03 FEC2           <1>         inc     dl      ; 3F9h (2F9h)     ; register for most significant byte
                        <1>                                     ; of the divisor value
00012F05 EE             <1>         out     dx, al ; 0
00012F06 EB00           <1>         JMP     $+2                 ; I/O DELAY
                        <1>         ;
00012F08 88E8           <1>         mov     al, ch ; 3          ; 8 data bits, 1 stop bit, no parity
                        <1>         ;and    al, 1Fh ; Bits 0,1,2,3,4
00012F0A 80C202         <1>         add     dl, 2  ; 3FBh (2FBh); Line control register
00012F0D EE             <1>         out     dx, al
00012F0E EB00           <1>         JMP     $+2                 ; I/O DELAY
                        <1>         ; 29/10/2015
00012F10 FECA           <1>         dec     dl     ; 3FAh (2FAh); FIFO Control register (16550/16750)
00012F12 30C0           <1>         xor     al, al              ; 0
00012F14 EE             <1>         out     dx, al              ; Disable FIFOs (reset to 8250 mode)
00012F15 EB00           <1>         JMP     $+2
                        <1> sp_i4:
                        <1> ;A18: ;----- COMM PORT STATUS ROUTINE
                        <1>         ; 29/06/2015 (line status after modem status)
00012F17 80C204         <1>         add     dl, 4  ; 3FEh (2FEh); Modem status register
                        <1> sp_i4s:
00012F1A EC             <1>         in      al, dx              ; GET MODEM CONTROL STATUS
00012F1B EB00           <1>         JMP     $+2                 ; I/O DELAY
00012F1D 88C4           <1>         mov     ah, al              ; PUT IN (AH) FOR RETURN
00012F1F FECA           <1>         dec     dl     ; 3FDh (2FDh); POINT TO LINE STATUS REGISTER
                        <1>                                     ; dx = 3FDh for COM1, 2FDh for COM2
00012F21 EC             <1>         in      al, dx              ; GET LINE CONTROL STATUS
                        <1>         ; AL = Line status, AH = Modem status
00012F22 C3             <1>         retn
                        <1>
                        <1> sp_status:
                        <1>         ; 29/06/2015
                        <1>         ; 27/06/2015 (Retro UNIX 386 v1)
                        <1>         ; Get serial port status
00012F23 66BAFE03       <1>         mov     dx, 3FEh            ; Modem status register (COM1)
00012F27 28C6           <1>         sub     dh, al              ; dh = 2 for COM2 (al = 1)
                        <1>                                     ; dx = 2FEh for COM2
```

```
00012F29 EBEF                <1>       jmp    short sp_i4s
                             <1>
                             <1> sp_setp: ; Set serial port communication parameters
                             <1>       ; 07/11/2015
                             <1>       ; 29/10/2015
                             <1>       ; 29/06/2015
                             <1>       ; Retro UNIX 386 v1 feature only !
                             <1>       ;
                             <1>       ; INPUT:
                             <1>       ;     AL = 0 for COM1
                             <1>       ;          1 for COM2
                             <1>       ;     AH = Communication parameters (*)
                             <1>       ; OUTPUT:
                             <1>       ;     CL = Line status
                             <1>       ;     CH = Modem status
                             <1>       ;   If cf = 1 -> Error code in [u.error]
                             <1>       ;             'invalid parameter !'
                             <1>       ;                   or
                             <1>       ;             'device not ready !' error
                             <1>       ;
                             <1>       ;  (*) Communication parameters (except BAUD RATE):
                             <1>       ;     Bit    4     3     2     1     0
                             <1>       ;            -PARITY--   STOP BIT  -WORD LENGTH-
                             <1>       ;  this one -->    00 = none    0 = 1 bit  11 = 8 bits
                             <1>       ;            01 = odd     1 = 2 bits   10 = 7 bits
                             <1>       ;            11 = even
                             <1>       ;  Baud rate setting bits: (29/06/2015)
                             <1>       ;            Retro UNIX 386 v1 feature only !
                             <1>       ;     Bit    7    6    5  | Baud rate
                             <1>       ;            ------------------------
                             <1>       ;     value  0    0    0  | Default (Divisor = 1)
                             <1>       ;            0    0    1  | 9600 (12)
                             <1>       ;            0    1    0  | 19200 (6)
                             <1>       ;            0    1   1  | 38400 (3)
                             <1>       ;            1    0   0  | 14400 (8)
                             <1>       ;            1    0   1  | 28800 (4)
                             <1>       ;            1    1    0  | 57600 (2)
                             <1>       ;            1    1    1  | 115200 (1)
                             <1>       ;
                             <1>       ; (COM1 base port address = 3F8h, COM1 Interrupt = IRQ 4)
                             <1>       ; (COM2 base port address = 2F8h, COM1 Interrupt = IRQ 3)
                             <1>       ;
                             <1>       ; ((Modified registers: EAX, ECX, EDX, EBX))
                             <1>       ;
00012F2B 66BAF803            <1>       mov    dx, 3F8h
00012F2F BB[7E8A0100]        <1>       mov    ebx, com1p ; COM1 control byte offset
00012F34 3C01                <1>       cmp    al, 1
00012F36 776B                <1>       ja     short sp_invp_err
00012F38 7203                <1>       jb     short sp_setp1 ;  COM1 (AL = 0)
00012F3A FECE                <1>       dec    dh ; 2F8h
00012F3C 43                  <1>       inc    ebx ; COM2 control byte offset
                             <1> sp_setp1:
                             <1>       ; 29/10/2015
00012F3D 8823                <1>       mov    [ebx], ah
00012F3F 0FB6CC              <1>       movzx  ecx, ah
00012F42 C0E905              <1>       shr    cl, 5 ; -> baud rate index
00012F45 80E41F              <1>       and    ah, 1Fh ; communication parameters except baud rate
00012F48 8A81[B22F0100]      <1>       mov    al, [ecx+b_div_tbl]
00012F4E 6689C1              <1>       mov    cx, ax
00012F51 E896FFFFFF          <1>       call   sp_i3
00012F56 6689C1              <1>       mov    cx, ax ; CL = Line status, CH = Modem status
00012F59 A880                <1>       test   al, 80h
00012F5B 740F                <1>       jz     short sp_setp2
00012F5D C603E3              <1>         mov     byte [ebx], 0E3h ; Reset to initial value (11100011b)
                             <1> stp_dnr_err:
00012F60 C705[C8030300]0F00- <1>       mov    dword [u.error], ERR_DEV_NOT_RDY ; 'device not ready !'
00012F68 0000                <1>
                             <1>       ; CL = Line status, CH = Modem status
00012F6A F9                  <1>       stc
00012F6B C3                  <1>       retn
                             <1> sp_setp2:
00012F6C 80FE02              <1>       cmp    dh, 2 ; COM2 (2F?h)
00012F6F 0F8649FFFFFF        <1>         jna     sp_i6
                             <1>                   ; COM1 (3F?h)
                             <1> sp_i5:
                             <1>       ; 07/11/2015
                             <1>       ; 26/10/2015
                             <1>       ; 29/06/2015
                             <1>       ;
                             <1>       ;; COM1 - enabling IRQ 4
                             <1>       pushf
00012F75 9C                  <1>       cli
00012F76 FA                  <1>       mov    dx, 3FCh            ; modem control register
00012F77 66BAFC03            <1>       in     al, dx                   ; read register
00012F7B EC                  <1>       JMP    $+2                 ; I/O DELAY
00012F7C EB00                <1>       or     al, 8               ; enable bit 3 (OUT2)
00012F7E 0C08                <1>       out    dx, al              ; write back to register
00012F80 EE                  <1>       JMP    $+2                 ; I/O DELAY
00012F81 EB00                <1>       mov    dx, 3F9h            ; interrupt enable register
00012F83 66BAF903            <1>       in     al, dx              ; read register
00012F87 EC                  <1>       JMP    $+2                 ; I/O DELAY
00012F88 EB00                <1>       ;or    al, 1               ; receiver data interrupt enable and
                             <1>       or     al, 3               ; transmitter empty interrupt enable
00012F8A 0C03                <1>       out    dx, al                   ; write back to register
00012F8C EE                  <1>       JMP    $+2                 ; I/O DELAY
00012F8D EB00                <1>       in     al, 21h             ; read interrupt mask register
00012F8F E421                <1>       JMP    $+2                 ; I/O DELAY
00012F91 EB00                <1>       and    al, 0EFh            ; enable IRQ 4 (COM1)
00012F93 24EF                <1>       out    21h, al             ; write back to register
00012F95 E621                <1>       ;
                             <1>       ; 23/10/2015
00012F97 B8[A72D0100]        <1>       mov    eax, com1_int
00012F9C A3[B92F0100]        <1>       mov    [com1_irq4], eax
                             <1>       ; 26/10/2015
00012FA1 9D                  <1>       popf
```

```
00012FA2 C3                    <1>       retn
                               <1>
                               <1> sp_invp_err:
00012FA3 C705[C8030300]1700-   <1>       mov    dword [u.error], ERR_INV_PARAMETER ; 'invalid parameter !'
00012FAB 0000                  <1>
00012FAD 31C9                  <1>       xor    ecx, ecx
00012FAF 49                    <1>       dec    ecx ; 0FFFFh
00012FB0 F9                    <1>       stc
00012FB1 C3                    <1>       retn
                               <1>
                               <1> ; 29/10/2015
                               <1> b_div_tbl: ; Baud rate divisor table (115200/divisor)
00012FB2 010C0603080401        <1>       db 1, 12, 6, 3, 8, 4, 1
                               <1>
                               <1>
                               <1> ; 23/10/2015
                               <1> com1_irq4:
00012FB9 [C12F0100]            <1>       dd dummy_retn
                               <1> com2_irq3:
00012FBD [C12F0100]            <1>       dd dummy_retn
                               <1>
                               <1> dummy_retn:
00012FC1 C3                    <1>       retn
                               <1>
                               <1> wakeup:
                               <1>       ; 24/01/2016
00012FC2 C3                    <1>       retn
                               <1>
                               <1> set_working_path_x:
                               <1>             ; 17/10/2016 (TRDOS 386 - FFF & FNF)
00012FC3 66B80100              <1>             mov    ax, 1
                               <1>                    ; File name is needed/forced (AL=1)
                               <1>                    ; Change directory as temporary (AH=0)
                               <1>
                               <1> set_working_path_xx: ; 30/12/2017 (syschdir)
                               <1>             ; This is needed for preventing wrong Find Next File
                               <1>             ; system call after sysopen, syscreate, sysmkdir etc.
                               <1>             ; Find Next File must immediate follow Find First File)
                               <1>
00012FC7 8825[CC960100]        <1>             mov    [FFF_Valid], ah ; 0 ; reset ; 17/10/2016
                               <1>
                               <1> set_working_path:
                               <1>             ; 16/10/2016
                               <1>             ; 12/10/2016
                               <1>             ; 10/10/2016
                               <1>             ; 05/10/2016 - TRDOS 386 (TRDOS v2.0)
                               <1>             ;
                               <1>             ; TRDOS v1.0 (DIR.ASM, "proc_set_working_path")
                               <1>                ; 27/01/2011 - 08/02/2011
                               <1>             ; Set/Changes current drive, directory and file
                               <1>             ; depending on command tail
                               <1>             ; (procedure is derivated from CMD_INTR.ASM
                               <1>             ; file or dir locating code of internal commands)
                               <1>             ; (This procedure is prepared for INT 21H file/dir
                               <1>             ; functions and also to get compact code for
                               <1>             ; internal mainprog -command interpreter- commands)
                               <1>             ;
                               <1>             ; INPUT: DS:SI -> Command tail (ASCIIZ string)
                               <1>             ; AL = 0 -> any, AL > 0 -> file name is forced
                               <1>             ; AH = CD -> Change directory permanently
                               <1>             ; AH <> CD -> Change directory as temporary
                               <1>             ;
                               <1>             ; OUTPUT: ES=DS, FindFile structure has been set
                               <1>             ;       RUN_CDRV points previous current drive
                               <1>             ;       DS:SI = FindFile structure address
                               <1>             ;       (DS=CS)
                               <1>             ;       AX, BX, CX, DX, DI will be changed
                               <1>             ;   cf = 1 -> Error code in AX (AL)
                               <1>             ;       stc & AX = 0 -> Bad command or path name
                               <1>             ; ---------------------------------------------
                               <1>             ;
                               <1>             ; TRDOS 386 (05/10/2016)
                               <1>             ; INPUT:
                               <1>             ;     ESI = File/Directory Path (ASCIIZ string)
                               <1>             ;             address in user's memory space
                               <1>             ;       AL = 0 -> any
                               <1>             ;       AL > 0 -> file name is forced
                               <1>             ;       AH = CD -> change directory as permanent
                               <1>             ;       AH <> CD -> change directory as temporary
                               <1>             ;
                               <1>             ; OUTPUT:
                               <1>             ;     FindFile structure has been set
                               <1>             ;       RUN_CDRV points previous current drive
                               <1>             ;       ESI = FindFile_Name address ; 12/10/2016
                               <1>             ;
                               <1>             ;       cf = 1 -> Error code in EAX (AL)
                               <1>             ;       stc & EAX = 0 -> Bad command or path name
                               <1>             ;
                               <1>             ; Modified registers: EAX, EBX, ECX, EDX, ESI, EDI
                               <1>
00012FCD 66A3[D0960100]        <1>             mov    [SWP_Mode], ax
00012FD3 A0[DE8A0100]          <1>             mov    al, [Current_Drv]
00012FD8 30E4                  <1>             xor    ah, ah
00012FDA 66A3[D2960100]        <1>             mov    [SWP_DRV], ax
                               <1>
                               <1>             ; TRDOS 386 ring 3 (user's page directory)
                               <1>             ; to ring 0 (kernel's page directory)
                               <1>             ; transfer modifications (05/10/2016).
                               <1>
00012FE0 55                    <1>             push   ebp
00012FE1 89E5                  <1>             mov    ebp, esp
                               <1>
00012FE3 B980000000            <1>             mov    ecx, 128 ; maximum path length = 128 bytes
00012FE8 29CC                  <1>             sub    esp, ecx ; reserve 128 bytes (buffer) on stack
00012FEA 89E7                  <1>             mov    edi, esp ; destination address (kernel space)
```

```
                                <1>             ; esi = source address (virtual, in user's memory space)
00012FEC E8A0EBFFFF             <1>             call   transfer_from_user_buffer
00012FF1 720A                   <1>             jc     short loc_swp_xor_retn
                                <1>
00012FF3 89E6                   <1>             mov    esi, esp ; temporary buffer (the path) on stack
                                <1> loc_swp_fchar:
00012FF5 8A06                   <1>             mov    al, [esi]
00012FF7 3C20                   <1>             cmp    al, 20h
00012FF9 7711                   <1>             ja     short loc_swp_parse_path_name
00012FFB 740C                   <1>             je     short loc_swp_fchar_next
                                <1>
                                <1> loc_swp_xor_retn:
00012FFD 31C0                   <1>             xor    eax, eax
00012FFF F9                     <1>             stc
                                <1> loc_swp_retn:
00013000 89EC                   <1>             mov    esp, ebp
00013002 5D                     <1>             pop    ebp
                                <1>
                                <1>             ;mov   esi, FindFile_Drv
00013003 BE[C0930100]           <1>             mov    esi, FindFile_Name ; 12/10/2016
00013008 C3                     <1>             retn
                                <1>
                                <1> loc_swp_fchar_next:
00013009 46                     <1>             inc    esi
0001300A EBE9                   <1>             jmp    short loc_swp_fchar
                                <1>
                                <1> loc_swp_parse_path_name:
0001300C BF[7E930100]           <1>             mov    edi, FindFile_Drv
00013011 E84C86FFFF             <1>             call   parse_path_name
00013016 72E8                   <1>             jc     short loc_swp_retn
                                <1>
                                <1> loc_swp_checkfile_name:
00013018 803D[D0960100]00       <1>             cmp    byte [SWP_Mode], 0
0001301F 761E                   <1>             jna    short loc_swp_drv
                                <1>
                                <1>             ; 10/10/2016 (valid file name checking)
00013021 BE[C0930100]           <1>             mov    esi, FindFile_Name
00013026 803E20                 <1>             cmp    byte [esi], 20h
00013029 76D2                   <1>             jna    short loc_swp_xor_retn
                                <1>
                                <1>             ; 16/10/2016
0001302B C605[CF960100]00       <1>             mov    byte [SWP_inv_fname], 0 ; reset
                                <1>             ; esi = file name address (ASCIIZ)
00013032 E81868FFFF             <1>             call   check_filename
00013037 7306                   <1>             jnc    short loc_swp_drv
                                <1>
00013039 FE05[CF960100]         <1>             inc    byte [SWP_inv_fname] ; set
                                <1> loc_swp_drv:
0001303F 8A35[DE8A0100]         <1>             mov    dh, [Current_Drv]
                                <1>               ;mov      [RUN_CDRV], dh
                                <1>
00013045 8A15[7E930100]         <1>             mov    dl, [FindFile_Drv]
                                <1>               ;cmp      dl, dh
0001304B 3A15[DE8A0100]         <1>             cmp    dl, [Current_Drv]
00013051 740D                   <1>             je     short loc_swp_change_directory
                                <1>
00013053 FE05[D3960100]         <1>             inc    byte [SWP_DRV_chg]
00013059 E89050FFFF             <1>             call   change_current_drive
0001305E 72A0                   <1>             jc     short loc_swp_retn ; eax = error code
                                <1>             ; eax = 0
                                <1>
                                <1> loc_swp_change_directory:
00013060 803D[7F930100]21       <1>             cmp    byte [FindFile_Directory], 21h
00013067 F5                     <1>             cmc
00013068 7396                   <1>             jnc    short loc_swp_retn
                                <1>
0001306A FE05[D3960100]         <1>             inc    byte [SWP_DRV_chg]
00013070 FE05[A6400100]         <1>             inc    byte [Restore_CDIR]
00013076 BE[7F930100]           <1>             mov    esi, FindFile_Directory
0001307B 8A25[D1960100]         <1>             mov    ah, [SWP_Mode+1]
00013081 E8C67FFFFF             <1>             call   change_current_directory
00013086 0F8274FFFFFF           <1>             jc     loc_swp_retn ; eax = error code
                                <1>
                                <1> loc_swp_change_prompt_dir_string:
                                <1>             ; esi = PATH_Array
                                <1>             ; eax = Current Directory First Cluster
                                <1>             ; edi = Logical DOS Drive Description Table
0001308C E8E07EFFFF             <1>             call   change_prompt_dir_str
00013091 29C0                   <1>             sub    eax, eax ; 0
00013093 E968FFFFFF             <1>             jmp    loc_swp_retn
                                <1>
                                <1> reset_working_path:
                                <1>             ; 06/10/2016 - TRDOS 386 (TRDOS v2.0)
                                <1>             ;
                                <1>             ; TRDOS v1.0 (DIR.ASM, "proc_reset_working_path")
                                <1>             ; 05/02/2011 - 08/02/2011
                                <1>             ;
                                <1>             ; Restores current drive and directory
                                <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>
00013098 31C0                   <1>             xor    eax, eax
0001309A 48                     <1>             dec    eax
                                <1>
0001309B 668B15[D2960100]       <1>             mov    dx, [SWP_DRV]
000130A2 08F6                   <1>             or     dh, dh
000130A4 742E                   <1>             jz     short loc_rwp_return
                                <1>
```

```
1040 000130A6 3A15[DE8A0100]      <1>             cmp    dl, [Current_Drv]
1041 000130AC 7407                <1>             je     short loc_rwp_restore_cdir
1042                              <1> loc_rwp_restore_cdrv:
1043 000130AE E83B50FFFF          <1>             call   change_current_drive
1044 000130B3 EB10                <1>             jmp    short loc_rwp_restore_ok
1045                              <1> loc_rwp_restore_cdir:
1046 000130B5 31DB                <1>             xor    ebx, ebx
1047 000130B7 88D7                <1>             mov    bh, dl
1048 000130B9 BE00010900          <1>             mov    esi, Logical_DOSDisks
1049 000130BE 01DE                <1>             add    esi, ebx
1050                              <1>
1051 000130C0 E8E050FFFF          <1>             call   restore_current_directory
1052                              <1>
1053                              <1> loc_rwp_restore_ok:
1054 000130C5 668B15[D2960100]    <1>             mov    dx, [SWP_DRV]
1055 000130CC 31C0                <1>             xor    eax, eax
1056 000130CE 66A3[D3960100]      <1>             mov    [SWP_DRV_chg], ax
1057                              <1> loc_rwp_return:
1058 000130D4 C3                  <1>             retn
1059                              <1>
1060                              <1> get_file_name:
1061                              <1>             ; 15/10/2016 - TRDOS 386 (TRDOS v2.0)
1062                              <1>             ; Convert file name
1063                              <1>             ;      from directory entry format
1064                              <1>                ;  to (8.3) dot file name format
1065                              <1>             ;
1066                              <1>             ; TRDOS v1.0 (DIR.ASM, "get_file_name")
1067                              <1>                ; 2005 - 09/10/2011
1068                              <1>             ; INPUT:
1069                              <1>             ;      DS:SI -> Directory Entry Format File Name
1070                              <1>             ;       ES:DI -> DOS Dot File Name Address
1071                              <1>             ; OUTPUT:
1072                              <1>             ;      DS:SI -> DOS Dot File Name Address
1073                              <1>                ;  ES:DI -> Directory Entry Format File Name
1074                              <1>             ;
1075                              <1>             ; TRDOS 386 (15/10/2016)
1076                              <1>             ; INPUT:
1077                              <1>             ;      ESI = File name addr in dir entry format
1078                              <1>             ;      EDI = Dot file name address (destination)
1079                              <1>             ; OUTPUT:
1080                              <1>             ;      File name is converted and moved
1081                              <1>             ;      to destination (as 8.3 dot filename)
1082                              <1>             ;
1083                              <1>             ; Modified registers: EAX, ECX
1084                              <1>
1085                              <1>                ; 2005 (TRDOS 8086) - 2016 (TRDOS 386)
1086                              <1>
1087 000130D5 57                  <1>             push   edi
1088 000130D6 56                  <1>             push   esi
1089 000130D7 AC                  <1>             lodsb
1090 000130D8 3C20                <1>             cmp    al, 20h
1091 000130DA 762A                <1>             jna    short pass_gfn_ext
1092 000130DC 56                  <1>             push   esi
1093 000130DD AA                  <1>             stosb
1094 000130DE B907000000          <1>             mov    ecx, 7
1095                              <1> loc_gfn_next_char:
1096 000130E3 AC                  <1>             lodsb
1097 000130E4 3C20                <1>             cmp    al, 20h
1098 000130E6 7603                <1>             jna    short pass_gfn_fn
1099 000130E8 AA                  <1>             stosb
1100 000130E9 E2F8                <1>             loop   loc_gfn_next_char
1101                              <1> pass_gfn_fn:
1102 000130EB 5E                  <1>             pop    esi
1103 000130EC 83C607              <1>             add    esi, 7
1104 000130EF AC                  <1>             lodsb
1105 000130F0 3C20                <1>             cmp    al, 20h
1106 000130F2 7612                <1>             jna    short pass_gfn_ext
1107 000130F4 B42E                <1>             mov    ah, '.'
1108 000130F6 86E0                <1>             xchg   ah, al
1109 000130F8 66AB                <1>             stosw
1110 000130FA AC                  <1>             lodsb
1111 000130FB 3C20                <1>             cmp    al, 20h
1112 000130FD 7607                <1>             jna    short pass_gfn_ext
1113 000130FF AA                  <1>             stosb
1114 00013100 AC                  <1>             lodsb
1115 00013101 3C20                <1>             cmp    al, 20h
1116 00013103 7601                <1>             jna    short pass_gfn_ext
1117 00013105 AA                  <1>             stosb
1118                              <1> pass_gfn_ext:
1119 00013106 30C0                <1>             xor    al, al
1120 00013108 AA                  <1>             stosb
1121 00013109 5E                  <1>             pop    esi
1122 0001310A 5F                  <1>             pop    edi
1123 0001310B C3                  <1>             retn
1124                              <1>
1125                              <1> set_hardware_int_vector:
1126                              <1>             ; 18/03/2017
1127                              <1>             ; 03/03/2017
1128                              <1>             ; 28/02/2017 - TRDOS 386 (TRDOS v2.0)
1129                              <1>             ;
1130                              <1>             ; SET/RESET HARDWARE INTERRUPT GATE
1131                              <1>             ;
1132                              <1>             ; Changes interrupt gate descriptor table
1133                              <1>             ; (without changing default interrupt list)
1134                              <1>             ;
1135                              <1>             ; INPUT:
1136                              <1>             ;      AL = IRQ number (0 to 15)
1137                              <1>             ;      AH > 0 -> set
1138                              <1>             ;      AH = 0 -> reset
1139                              <1>             ;
1140                              <1>             ; Modified registers: eax, ebx, edx, edi
1141                              <1>             ;
1142                              <1>
1143 0001310C C0E002              <1>             shl    al, 2 ; IRQ number * 4
1144 0001310F 0FB6D8              <1>             movzx  ebx, al
```

```
1145                            <1>
1146 00013112 08E4              <1>             or      ah, ah
1147 00013114 7508              <1>             jnz     short shintv_1 ; set (for user call service)
1148                            <1>
1149                            <1>             ; 18/03/2017
1150 00013116 81C3[A44A0100]    <1>             add     ebx, IRQ_list ; reset to default interrupt list
1151 0001311C EB06              <1>             jmp     short shintv_2
1152                            <1> shintv_1:
1153 0001311E 81C3[45310100]    <1>             add     ebx, IRQ_u_list
1154                            <1> shintv_2:
1155 00013124 8B13              <1>             mov     edx, [ebx] ; IRQ handler address
1156                            <1>
1157                            <1>             ; 03/03/2017
1158 00013126 D0E0              <1>             shl     al, 1 ; IRQ number * 8
1159                            <1>             ; 18/03/2017
1160 00013128 0FB6F8            <1>             movzx   edi, al
1161 0001312B 81C7[30880100]    <1>             add     edi, idt + (8*32) ; IRQ 0 offset = idt + 256
1162                            <1>
1163 00013131 89D0              <1>             mov     eax, edx ; IRQ handler address
1164 00013133 BB00000800        <1>             mov     ebx, 80000h
1165                            <1>
1166                            <1>             ;mov    edx, eax
1167 00013138 66BA008E          <1>             mov     dx, 8E00h
1168 0001313C 6689C3            <1>             mov     bx, ax
1169 0001313F 89D8              <1>             mov     eax, ebx ; /* selector = 0x0008 = cs */
1170                            <1>                                      ; /* interrupt gate - dpl=0, present */
1171 00013141 AB                <1>             stosd   ; selector & offset bits 0-15
1172 00013142 8917              <1>             mov     [edi], edx ; attributes & offset bits 16-23
1173                            <1>
1174 00013144 C3                <1>             retn
1175                            <1> IRQ_u_list:
1176                            <1>             ; 28/02/2017
1177 00013145 [5E090000]        <1>             dd      timer_int
1178 00013149 [D6100000]        <1>             dd      kb_int
1179 0001314D [400B0000]        <1>             dd      irq2
1180 00013151 [85310100]        <1>             dd      IRQ_service3
1181 00013155 [8F310100]        <1>             dd      IRQ_service4
1182 00013159 [99310100]        <1>             dd      IRQ_service5
1183 0001315D [DB510000]        <1>             dd      fdc_int
1184 00013161 [A3310100]        <1>             dd      IRQ_service7
1185 00013165 [C90A0000]        <1>             dd      rtc_int
1186 00013169 [AD310100]        <1>             dd      IRQ_service9
1187 0001316D [B7310100]        <1>             dd      IRQ_service10
1188 00013171 [C1310100]        <1>             dd      IRQ_service11
1189 00013175 [CB310100]        <1>             dd      IRQ_service12
1190 00013179 [D5310100]        <1>             dd      IRQ_service13
1191 0001317D [8E5B0000]        <1>             dd      hdc1_int
1192 00013181 [B55B0000]        <1>             dd      hdc2_int
1193                            <1>
1194                            <1>             ; 03/03/2017
1195                            <1>             ; 27/02/2017
1196                            <1> IRQ_service3:
1197 00013185 36C605[969C0100]03 <1>             mov     byte [ss:IRQnum], 3
1198 0001318D EB4E              <1>             jmp     short IRQ_service
1199                            <1> IRQ_service4:
1200 0001318F 36C605[969C0100]04 <1>             mov     byte [ss:IRQnum], 4
1201 00013197 EB44              <1>             jmp     short IRQ_service
1202                            <1> IRQ_service5:
1203 00013199 36C605[969C0100]05 <1>             mov     byte [ss:IRQnum], 5
1204 000131A1 EB3A              <1>             jmp     short IRQ_service
1205                            <1> IRQ_service7:
1206 000131A3 36C605[969C0100]07 <1>             mov     byte [ss:IRQnum], 7
1207 000131AB EB30              <1>             jmp     short IRQ_service
1208                            <1> IRQ_service9:
1209 000131AD 36C605[969C0100]09 <1>             mov     byte [ss:IRQnum], 9
1210 000131B5 EB26              <1>             jmp     short IRQ_service
1211                            <1> IRQ_service10:
1212 000131B7 36C605[969C0100]0A <1>             mov     byte [ss:IRQnum], 10
1213 000131BF EB1C              <1>             jmp     short IRQ_service
1214                            <1> IRQ_service11:
1215 000131C1 36C605[969C0100]0B <1>             mov     byte [ss:IRQnum], 11
1216 000131C9 EB12              <1>             jmp     short IRQ_service
1217                            <1> IRQ_service12:
1218 000131CB 36C605[969C0100]0C <1>             mov     byte [ss:IRQnum], 12
1219 000131D3 EB08              <1>             jmp     short IRQ_service
1220                            <1> IRQ_service13:
1221 000131D5 36C605[969C0100]0D <1>             mov     byte [ss:IRQnum], 13
1222                            <1>             ;jmp    short IRQ_service
1223                            <1> IRQ_service:
1224                            <1>             ; 13/06/2017
1225                            <1>             ; 11/06/2017
1226                            <1>             ; 10/06/2017
1227                            <1>             ; 01/03/2017, 04/03/2017
1228                            <1>             ; 27/02/2017, 28/02/2017
1229 000131DD 1E                <1>             push    ds
1230 000131DE 06                <1>             push    es
1231 000131DF 0FA0              <1>             push    fs
1232 000131E1 0FA8              <1>             push    gs
1233                            <1>
1234 000131E3 60                <1>             pushad ; eax,ecx,edx,ebx,esp,ebp,esi,edi
1235 000131E4 66B91000          <1>             mov     cx, KDATA
1236 000131E8 8ED9              <1>             mov     ds, cx
1237 000131EA 8EC1              <1>             mov     es, cx
1238 000131EC 8EE1              <1>             mov     fs, cx
1239 000131EE 8EE9              <1>             mov     gs, cx
1240                            <1>
1241 000131F0 0F20D8            <1>             mov     eax, cr3
1242 000131F3 A3[929C0100]      <1>             mov     [IRQ_cr3], eax
1243                            <1>
1244 000131F8 A1[188A0100]      <1>             mov     eax, [k_page_dir]
1245 000131FD 0F22D8            <1>             mov     cr3, eax
1246                            <1>
1247 00013200 A0[969C0100]      <1>             mov     al, [IRQnum]
1248                            <1>
1249                            <1>             ;mov    cl, [sysflg]
```

```
1250                            <1>             ;mov   [u.r_mode], cl  ; system (0) or user mode (FFh)
1251                            <1> IRQsrv_0:
1252 00013205 0FB6D8           <1>             movzx  ebx, al
1253 00013208 8A9B[DA490100]    <1>             mov    bl, [ebx+IRQenum] ; IRQ (available) index number + 1
1254                            <1>             ; 01/03/2017
1255 0001320E FECB              <1>             dec    bl  ; IRQ index number, 0 to 8
1256 00013210 0F8807010000      <1>             js     IRQsrv_5 ; not available to use here!?
1257                            <1>             ;
1258 00013216 80BB[5C9C0100]80  <1>             cmp    byte [ebx+IRQ.method], 80h ; using by a dev or kernel?
1259 0001321D 7205              <1>             jb     short IRQsrv_1 ; no
1260                            <1>
1261                            <1>             ; If the IRQ service is already owned by TRDOS 386 kernel
1262                            <1>             ;      or a Device driver
1263                            <1>             ; we need to call 'dev_IRQ_service'
1264                            <1>
1265                            <1>             ; IRQ number in AL
1266 0001321F E866020000          <1>             call   dev_IRQ_service      ; IRQ service for device drivers
1267                            <1>             ; IRQ number in AL
1268                            <1> IRQsrv_1:
1269                            <1>             ; check user callback service status
1270                            <1>             ; AL = IRQ number
1271                            <1>             ; EBX = IRQ (Available) Index number
1272                            <1>
1273 00013224 A2[D7030300]      <1>             mov    [u.irqwait], al ; set waiting IRQ flag
1274                            <1>
1275 00013229 8A83[4A9C0100]    <1>             mov    al, [ebx+IRQ.owner]
1276 0001322F 20C0              <1>             and    al, al
1277 00013231 0F84E6000000      <1>             jz     IRQsrv_5 ; it is not owned by a user/proc
1278                            <1>
1279                            <1>             ; 03/03/2017
1280 00013237 89DA              <1>             mov    edx, ebx
1281 00013239 C0E202            <1>             shl    dl, 2
1282 0001323C 8B92[6E9C0100]    <1>             mov    edx, [edx+IRQ.addr] ; S.R.B. or Callback service addr
1283                            <1>
1284 00013242 8AA3[5C9C0100]    <1>             mov    ah, [ebx+IRQ.method]
1285 00013248 F6C401            <1>             test   ah, 1
1286 0001324B 7534              <1>             jnz    short IRQsrv_4 ; Callback service method
1287                            <1>
1288                            <1>             ; Signal Response Byte method
1289                            <1>             ;mov   edx, [edx+IRQ.addr] ; Signal Response Byte address
1290                            <1>             ;                          ; (Physical address, non-swappable)
1291 0001324D 80E402            <1>             and    ah, 2 ; bit 1, (S.R.B.) counter (auto increment) method
1292 00013250 8AA3[659C0100]    <1>             mov    ah, [ebx+IRQ.srb] ; Signal Response Byte value
1293 00013256 7408              <1>             jz     short IRQsrv_2 ; fixed S.R.B. value
1294                            <1>             ; counter method (auto increment)
1295 00013258 FEC4              <1>             inc    ah
1296 0001325A 88A3[659C0100]    <1>             mov    [ebx+IRQ.srb], ah ; next (count) number
1297                            <1> IRQsrv_2:
1298 00013260 8822              <1>             mov    [edx], ah ; put S.R.B. val to the user's S.R.B. addr
1299 00013262 C605[D7030300]00  <1>             mov    byte [u.irqwait], 0 ; clear waiting IRQ flag
1300                            <1>
1301 00013269 3A05[B3030300]    <1>             cmp    al, [u.uno]
1302 0001326F 0F84A8000000      <1>             je     IRQsrv_5 ; the owner is current user/process
1303                            <1> IRQsrv_3:
1304                            <1>             ; the owner is not current user/process
1305                            <1>             ; AL = process number
1306 00013275 B202              <1>             mov    dl, 2 ; priority, 2 = event (high)
1307 00013277 E837FAFFFF        <1>             call   set_run_sequence
1308                            <1>
1309                            <1>             ; [u.irqwait] = waiting IRQ number for callback service
1310                            <1>
1311 0001327C E99C000000        <1>             jmp    IRQsrv_5
1312                            <1> IRQsrv_4:
1313 00013281 3A05[B3030300]    <1>             cmp    al, [u.uno]  ; is the owner is current user/process?
1314 00013287 75EC              <1>             jne    short IRQsrv_3 ; no !
1315                            <1>
1316                            <1>             ; Check if an IRQ callback service already in progress
1317 00013289 803D[D8030300]00  <1>             cmp    byte [u.r_lock], 0
1318 00013290 0F8787000000      <1>             ja     IRQsrv_5 ; nothing to do !
1319                            <1>                                ; (we need to complete prev callback)
1320 00013296 803D[D4030300]00  <1>             cmp    byte [u.t_lock], 0
1321 0001329D 777E              <1>             ja     short IRQsrv_5 ; nothing to do !
1322                            <1>                                ; (we need to complete timer callback)
1323                            <1>
1324                            <1>             ; 04/03/2017
1325 0001329F C605[D7030300]00  <1>             mov    byte [u.irqwait], 0 ; reset/clear waiting IRQ flag
1326                            <1>
1327 000132A6 FE05[D8030300]    <1>             inc    byte [u.r_lock] ; 'IRQ callback service in progress' flag
1328                            <1>
1329 000132AC 8A0D[5B030300]    <1>             mov    cl, [sysflg]   ; (system call) mode flag (kernel/user)
1330 000132B2 880D[D9030300]    <1>             mov    [u.r_mode], cl ; system mode (0) or user mode (FFh)
1331                            <1>
1332                            <1>             ;
1333 000132B8 8B2D[B4890100]    <1>             mov    ebp, [tss.esp0] ; kernel stack address (for ring 0)
1334 000132BE 83ED14            <1>             sub    ebp, 20             ; eip, cs, eflags, esp, ss
1335 000132C1 892D[5C030300]    <1>             mov    [u.sp], ebp
1336 000132C7 8925[60030300]    <1>             mov    [u.usp], esp
1337                            <1>
1338                            <1>             ;or    word [ebp+8], 200h ; 22/01/2017, force enabling interrupts
1339                            <1>
1340 000132CD 8B44241C          <1>             mov    eax, [esp+28] ; pushed eax
1341 000132D1 A3[64030300]      <1>             mov    [u.r0], eax
1342                            <1>
1343 000132D6 E820E7FFFF        <1>             call   wswap ; save user's registers & status
1344                            <1>
1345                            <1>             ; software int is in ring 0 but IRQ handler must return to ring 3
1346                            <1>             ; so, ring 3 return address and stack registers
1347                            <1>             ; (eip, cs, eflags, esp, ss)
1348                            <1>             ; must be copied to IRQ handler return
1349                            <1>             ; eip will be replaced by callback service routine address
1350                            <1>
1351 000132DB C605[5B030300]FF  <1>             mov    byte [sysflg], 0FFh ; user mode
1352                            <1>
1353                            <1>             ; system mode (system call)
1354                            <1>             ;mov   ebp, [u.sp] ; EIP (u), CS (UCODE), EFLAGS (u),
```

```
                                <1>                             ; ESP (u), SS (UDATA)
                                <1>
000132E2 8B4510                 <1>             mov     eax, [ebp+16]; SS (UDATA)
000132E5 89E6                   <1>             mov     esi, esp
000132E7 50                     <1>             push    eax
000132E8 50                     <1>             push    eax
000132E9 89E7                   <1>             mov     edi, esp
000132EB 893D[60030300]         <1>             mov     [u.usp], edi
000132F1 B908000000             <1>             mov     ecx, ((ESPACE/4) - 4) ; except DS, ES, FS, GS
000132F6 F3A5                   <1>             rep     movsd
000132F8 B104                   <1>             mov     cl, 4
000132FA F3AB                   <1>             rep     stosd
000132FC 893D[5C030300]         <1>             mov     [u.sp], edi
00013302 89EE                   <1>             mov     esi, ebp
00013304 B105                   <1>             mov     cl, 5 ; EIP (u), CS (UCODE), EFLAGS (u), ESP (u), SS (UDATA)
00013306 F3A5                   <1>             rep     movsd
                                <1>             ;
                                <1>
00013308 8B0D[B8030300]         <1>             mov     ecx, [u.pgdir]
0001330E 890D[929C0100]         <1>             mov     [IRQ_cr3], ecx
                                <1>
                                <1> set_IRQ_callback_addr:
                                <1>             ;
                                <1>             ; This routine sets return address
                                <1>             ; to start of user's interrupt
                                <1>             ; service (callback) address
                                <1>             ;
                                <1>             ; INPUT:
                                <1>             ;     EDX = callback routine/service address
                                <1>             ;           (virtual, not physical address!)
                                <1>             ;     [u.sp] = kernel stack, points to
                                <1>             ;              user's EIP,CS,EFLAGS,ESP,SS
                                <1>             ;              registers.
                                <1>             ; OUTPUT:
                                <1>             ;     EIP (user) = callback (service) address
                                <1>             ;     CS (user) = UCODE
                                <1>             ;     EFLAGS (user) = flags before callback
                                <1>             ;      ESP (user) = ESP-4 (user, before callback)
                                <1>             ;     [ESP](user) = EIP (user) before callback
                                <1>             ;
                                <1>             ; Note: If CPU was in user mode while entering
                                <1>             ;     the timer interrupt service routine,
                                <1>             ;     'IRET' will get return to callback routine
                                <1>             ;     immediately. If CPU was in system/kernel mode
                                <1>             ;     'iret' will get return to system call and
                                <1>             ;     then, callback routine will be return address
                                <1>             ;     from system call. (User's callback/service code
                                <1>             ;     will be able to return to normal return address
                                <1>             ;     via a 'sysrele' system call at the end.)
                                <1>             ;
                                <1>             ; Note: User's IRQ callback service code must be ended
                                <1>             ;     with a 'sysrele' system call !
                                <1>             ;
                                <1>             ;     For example:
                                <1>             ;
                                <1>             ;     audio_IRQ_callback:
                                <1>             ;         ...
                                <1>             ;         <load DMA buffer with audio data>
                                <1>             ;         ...
                                <1>             ;         mov eax, 39 ; 'sysrele'
                                <1>             ;         int 40h ; TRDOS 386 system call (interrupt)
                                <1>             ;
                                <1>
                                <1>             ;mov   edx, [edx+IRQ.addr] ; Callback service address
                                <1>             ;                       ; (Virtual address)
                                <1>
00013314 8B2D[5C030300]         <1>             mov     ebp, [u.sp]; kernel's stack, points to EIP (user)
0001331A 895500                 <1>             mov     [ebp], edx
                                <1> IRQsrv_5:
                                <1>             ; EOI & return
                                <1>             ; 01/08/2020
                                <1>             ; 11/06/2017
                                <1>             ; 10/06/2017
                                <1>             ;mov   al, [IRQnum]
0001331D B020                   <1>             mov     al, 20h ; 01/08/2020
0001331F FA                     <1>             cli
                                <1>             ;cmp   al, 7
00013320 803D[969C0100]07       <1>             cmp     byte [IRQnum], 7 ; 01/08/2020
00013327 7602                   <1>             jna     short IRQsrv_6
                                <1>             ;
                                <1>             ;;mov  al, EOI      ; end of interrupt
                                <1>             ;mov   al, 20h ; 01/08/2020
                                <1>             ;cli         ; disable interrupts till stack cleared
                                <1>             ;out   INTB00, al ; For controll2 #2
00013329 E6A0                   <1>             out     0A0h, al
                                <1> IRQsrv_6:
                                <1>             ;mov   byte [IRQnum], 0 ; reset
                                <1>             ;;mov  al, EOI      ; end of interrupt
                                <1>             ;mov   al, 20h ; 01/08/2020
                                <1>             ;cli         ; disable interrupts till stack cleared
                                <1>             ;out   INTA00, al ; end of interrupt to 8259 - 1
0001332B E620                   <1>             out     20h, al
                                <1> IRQsrv_7:
                                <1>             ;; 13/06/2017
                                <1>             ;or    word [ebp+8], 200h ; force enabling interrupts
                                <1>             ;
0001332D 8B0D[929C0100]         <1>             mov     ecx, [IRQ_cr3]      ; previous content of cr3 register
00013333 0F22D9                 <1>             mov     cr3, ecx     ; restore cr3 register content
                                <1>             ;
00013336 61                     <1>             popad ; edi,esi,ebp,(icrement esp by 4),ebx,edx,ecx,eax
                                <1>             ;
00013337 0FA9                   <1>             pop     gs
00013339 0FA1                   <1>             pop     fs
0001333B 07                     <1>             pop     es
0001333C 1F                     <1>             pop     ds
```

```
                                <1>             ;
0001333D 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>
0001333E BF[D5960100]           <1>             mov    edi, device_name
00013343 E805010000             <1>             call   lodsb_capitalize
00013348 88C4                   <1>             mov    ah, al
0001334A 3C2F                   <1>             cmp    al, '/'
0001334C 750E                   <1>             jne    short gdn_1
0001334E BF[D5960100]           <1>             mov    edi, device_name
00013353 E8F5000000             <1>             call   lodsb_capitalize
                                <1> gdn_0:
00013358 20C0                   <1>             and    al, al ; 0 ?
0001335A 7420                   <1>             jz     short gdn_err ; null name after '/'
                                <1> gdn_1:
0001335C 3C44                   <1>             cmp    al, 'D'
0001335E 7517                   <1>             jne    short gdn_2
00013360 E8E8000000             <1>             call   lodsb_capitalize
00013365 3C45                   <1>             cmp    al, 'E'
00013367 750E                   <1>             jne    short gdn_2
00013369 E8DF000000             <1>             call   lodsb_capitalize
0001336E 3C56                   <1>             cmp    al, 'V'
00013370 7505                   <1>             jne    short gdn_2
00013372 AC                     <1>             lodsb
00013373 3C2F                   <1>             cmp    al, '/'
00013375 740D                   <1>             je     short gdn_4
                                <1> gdn_2:
00013377 80FC2F                 <1>             cmp    ah, '/'
0001337A 750F                   <1>             jne    short gdn_5
                                <1> gdn_err:
                                <1>             ; invalid device name or device not found
0001337C 31C0                   <1>             xor    eax, eax ; 0
0001337E F9                     <1>             stc
0001337F C3                     <1>             retn
                                <1> gdn_3:
00013380 3C2F                   <1>             cmp    al, '/'
00013382 7507                   <1>             jne    short gdn_5
                                <1> gdn_4:
00013384 BF[D5960100]           <1>             mov    edi, device_name
00013389 EB04                   <1>             jmp    short gdn_6
                                <1> gdn_5:
0001338B 3C00                   <1>             cmp    al, 0
0001338D 7419                   <1>             je     short gdn_7
                                <1> gdn_6:
0001338F E8B9000000             <1>             call lodsb_capitalize
00013394 81FF[DD960100]         <1>             cmp    edi, device_name + 8
0001339A 72E4                   <1>             jb     short gdn_3
0001339C 3C00                   <1>             cmp    al, 0
0001339E 75DC                   <1>             jne    short gdn_err
000133A0 81FF[D6960100]         <1>             cmp    edi, device_name + 1
000133A6 76D4                   <1>             jna    short gdn_err ; null name after '/'
                                <1> gdn_7:
000133A8 AA                     <1>             stosb
                                <1>             ; zero padding ("NAME",0,0,0,0)
000133A9 81FF[DD960100]         <1>             cmp    edi, device_name + 8
000133AF 72F7                   <1>             jb     short gdn_7
                                <1> gdn_8:
                                <1>             ; search for kernel device names
000133B1 BE[D5960100]           <1>             mov    esi, device_name
000133B6 BF[C0470100]           <1>             mov    edi, KDEV_NAME
000133BB 31C0                   <1>             xor    eax, eax
                                <1> gdn_9:
000133BD A7                     <1>             cmpsd
000133BE 7505                   <1>             jne    short gdn_10
000133C0 A7                     <1>             cmpsd
000133C1 7503                   <1>             jne    short gdn_11
000133C3 EB2B                   <1>             jmp    short gdn_17 ; match
                                <1> gdn_10:
000133C5 A7                     <1>             cmpsd  ; add esi, 4 & add edi, 4
                                <1> gdn_11:
000133C6 BE[D5960100]           <1>             mov    esi, device_name
000133CB FEC0                   <1>             inc    al
000133CD 3C16                   <1>             cmp    al, NumOfKernelDevNames
```

```
1565 000133CF 72EC              <1>             jb      short gdn_9
1566                            <1> gdn_12:
1567                            <1>             ; search for installable device names
1568                            <1>             ; esi = offset device_name
1569 000133D1 BF[00970100]      <1>             mov     edi, IDEV_NAME
1570 000133D6 28C0              <1>             sub     al, al ; 0
1571                            <1> gdn_13:
1572 000133D8 A7                <1>             cmpsd
1573 000133D9 7505              <1>             jne     short gdn_14
1574 000133DB A7                <1>             cmpsd
1575 000133DC 7503              <1>             jne     short gdn_15
1576 000133DE EB3F              <1>             jmp     short gdn_19 ; match
1577                            <1> gdn_14:
1578 000133E0 A7                <1>             cmpsd  ; add esi, 4 & add edi, 4
1579                            <1> gdn_15:
1580 000133E1 BE[D5960100]      <1>             mov     esi, device_name
1581 000133E6 FEC0              <1>             inc     al
1582 000133E8 3C08              <1>             cmp     al, NumOfInstallableDevices
1583 000133EA 72EC              <1>             jb      short gdn_13
1584                            <1>
1585                            <1> gdn_16:     ; error: invalid device name (not found) !
1586 000133EC 30C0              <1>             xor     al, al
1587 000133EE F9                <1>             stc
1588 000133EF C3                <1>             retn
1589                            <1>
1590                            <1> gdn_17:             ; name match (with one of kernel device names)
1591                            <1>             ;
1592                            <1>             ; convert KDEV_NAME index to
1593                            <1>             ; KDEV_NUMBER index
1594                            <1>             ; (different names are used for same devices)
1595                            <1>             ; (example: "COM1" & "TTY8" = device number 18)
1596 000133F0 89C3              <1>             mov     ebx, eax ; < 256
1597 000133F2 8A83[70480100]    <1>             mov     al, [KDEV_NUMBER+ebx]
1598                            <1>
1599                            <1>             ; check if empty dev entry in the list
1600 000133F8 80B8[84980100]00  <1>             cmp     byte [DEV_OPENMODE+eax], 0
1601 000133FF 771B              <1>             ja      short gdn_18 ; it must be already set
1602                            <1>
1603                            <1>             ; (re)set device name and access flags
1604                            <1>             ; (remain open work will be easy after that)
1605                            <1>             ; (NOTE: here, data will be copied to bss section)
1606 00013401 88C3              <1>             mov     bl, al
1607 00013403 83EF08            <1>             sub     edi, 8 ; kernel device name address (data)
1608 00013406 66C1E302          <1>             shl     bx, 2
1609 0001340A 89BB[A2980100]    <1>             mov     [DEV_NAME_PTR+ebx], edi ; (all) device names
1610 00013410 8A98[C6490100]    <1>             mov     bl, [KDEV_ACCESS+eax] ; kernel dev list (data)
1611 00013416 8898[D0970100]    <1>             mov     [DEV_ACCESS+eax], bl ; (all) device list (bss)
1612                            <1> gdn_18:
1613 0001341C FEC0              <1>             inc     al ; 1 to NumOfKernelDevNames (<=7Fh)
1614                            <1>             ; eax = device index/entry number
1615 0001341E C3                <1>             retn
1616                            <1>
1617                            <1> gdn_19:             ; name match (with one of installable device names)
1618                            <1>             ;
1619                            <1>             ; al = 0 to NumOfInstallableDevices - 1 (<=7Fh)
1620                            <1>
1621 0001341F 89C3              <1>             mov     ebx, eax
1622 00013421 80C316            <1>             add     bl, NumOfKernelDevices    ; < NUMOFDEVICES
1623                            <1>
1624                            <1>             ; check if empty dev entry in the list
1625 00013424 80BB[84980100]00  <1>             cmp     byte [DEV_OPENMODE+ebx], 0
1626 0001342B 771D              <1>             ja      short gdn_20 ; it must be already set
1627                            <1>
1628                            <1>             ; (re)set device name and access flags
1629                            <1>             ; (remain open work will be easy after that)
1630 0001342D 83EF08            <1>             sub     edi, 8 ; installable device name address
1631 00013430 66C1E302          <1>             shl     bx, 2 ;*4
1632 00013434 89BB[A2980100]    <1>             mov     [DEV_NAME_PTR+ebx], edi ; (all) device names
1633 0001343A 66C1EB02          <1>             shr     bx, 2
1634 0001343E 8A80[48970100]    <1>             mov     al, [IDEV_FLAGS+eax] ; installable dev list
1635 00013444 8883[D0970100]    <1>             mov     [DEV_ACCESS+ebx], al ; (all) device list
1636                            <1> gdn_20:
1637 0001344A 88D8              <1>             mov     al, bl
1638                            <1>             ; eax = device index/entry number ; < NUMOFDEVICES
1639 0001344C C3                <1>             retn
1640                            <1>
1641                            <1> lodsb_capitalize:
1642                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
1643                            <1>       ; INPUT -> [esi] = character
1644                            <1>       ;          edi = destination
1645                            <1>       ; OUTPUT -> AL contains capitalized character
1646                            <1>       ;         esi = esi+1
1647                            <1>       ;         edi = edi+1
1648                            <1>       ;
1649 0001344D AC                <1>       lodsb
1650 0001344E 3C61              <1>       cmp al, 61h
1651 00013450 7206              <1>       jb  short lodsb_cap_retn
1652 00013452 3C7A              <1>       cmp al, 7Ah
1653 00013454 7702              <1>       ja  short lodsb_cap_retn
1654 00013456 24DF              <1>       and al, 0DFh
1655                            <1> lodsb_cap_retn:
1656 00013458 AA                <1>       stosb
1657 00013459 C3                <1>       retn
1658                            <1>
1659                            <1> device_open:
1660                            <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
1661                            <1>       ; Complete device opening work for sysopen (device)
1662                            <1>       ;
1663                            <1>       ; INPUT ->
1664                            <1>       ;      EAX = Device Number (AL)
1665                            <1>       ;         CL = Open mode (1 = read, 2 = write)
1666                            <1>       ;       CH = Device access byte (bit 0 = 0)
1667                            <1>       ; OUTPUT ->
1668                            <1>       ;      EAX = Device Number
1669                            <1>       ;      CF = 0 -> device has been opened
```

```
1670                            <1>       ;      CF = 1 -> device could not be opened
1671                            <1>       ;
1672                            <1>       ;  Modified registers: ebx, (edx, ecx, esi, edi, ebp)
1673                            <1>       ;
1674                            <1>
1675 0001345A 89C3             <1>       mov    ebx, eax
1676 0001345C 66C1E302         <1>       shl    bx, 2 ; *4
1677                            <1>
1678 00013460 F6C580           <1>       test   ch, 80h ; bit 7, installable device driver flag
1679 00013463 7406             <1>       jz     short d_open_2 ; Kernel device
1680                            <1>       ; installable device
1681                            <1> d_open_1:
1682 00013465 FFA3[4C970100]    <1>         jmp  dword [ebx+IDEV_OADDR-4]
1683                            <1> d_open_2:
1684 0001346B FFA3[82480100]    <1>       jmp    dword [ebx+KDEV_OADDR-4]
1685                            <1>
1686                            <1> device_close:
1687                            <1>       ; 08/10/2016 - TRDOS 386 (TRDOS v2.0)
1688                            <1>       ; Complete device closing work for sysclose (device)
1689                            <1>       ;
1690                            <1>       ; INPUT ->
1691                            <1>       ;      EAX = Device Number (AL)
1692                            <1>       ;        CL = Open mode (1 = read, 2 = write)
1693                            <1>       ;       CH = Device access byte (bit 0 = 0)
1694                            <1>       ; OUTPUT ->
1695                            <1>       ;      EAX = Device Number
1696                            <1>       ;      CF = 0 -> device has been closed
1697                            <1>       ;      CF = 1 -> device could not be closed
1698                            <1>       ;
1699                            <1>       ; Modified registers: ebx, (edx, ecx, esi, edi, ebp)
1700                            <1>       ;
1701                            <1>
1702 00013471 89C3             <1>       mov    ebx, eax
1703 00013473 66C1E302         <1>       shl    bx, 2 ; *4
1704                            <1>
1705 00013477 F6C580           <1>       test   ch, 80h ; bit 7, installable device driver flag
1706 0001347A 7406             <1>       jz     short d_close_2 ; Kernel device
1707                            <1>       ; installable device
1708                            <1> d_close_1:
1709 0001347C FFA3[6C970100]    <1>         jmp  dword [ebx+IDEV_CADDR-4]
1710                            <1> d_close_2:
1711 00013482 FFA3[D2480100]    <1>       jmp    dword [ebx+KDEV_CADDR-4]
1712                            <1>
1713                            <1> rnull:
1714                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
1715                            <1>       ; read null (read from null device)
1716 00013488 C3                 <1>       retn
1717                            <1>
1718                            <1> wnull:
1719                            <1>       ; 07/10/2016 - TRDOS 386 (TRDOS v2.0)
1720                            <1>       ; write null (write to null device)
1721 00013489 C3                 <1>       retn
1722                            <1>
1723                            <1> dev_IRQ_service:
1724                            <1>       ; 12/05/2017
1725                            <1>       ; 13/04/2017
1726                            <1>       ; 27/02/2017 - TRDOS 386 (TRDOS v2.0)
1727                            <1>       ; INPUT ->
1728                            <1>       ;      AL = IRQ Number (0 to 15)
1729                            <1>       ;
1730 0001348A 53                 <1>       push   ebx
1731 0001348B 0FB6D8             <1>       movzx  ebx, al
1732 0001348E C0E302             <1>       shl    bl, 2 ; * 4
1733 00013491 8B9B[0A9C0100]     <1>       mov    ebx, [ebx+DEV_INT_HNDLR]
1734 00013497 21DB               <1>       and    ebx, ebx
1735 00013499 7404               <1>         jz   short dIRQ_s_retn
1736 0001349B 50                 <1>       push   eax
1737                            <1>
1738 0001349C FFD3               <1>       call   ebx
1739                            <1>
1740 0001349E 58                 <1>       pop    eax
1741                            <1> dIRQ_s_retn:
1742 0001349F 5B                 <1>       pop    ebx
1743 000134A0 C3                 <1>       retn
1744                            <1>
1745                            <1>
1746                            <1> set_dev_IRQ_service:
1747                            <1>       ; 13/04/2017 - TRDOS 386 (TRDOS v2.0)
1748                            <1>       ;
1749                            <1>       ; Set Device Interrupt Service
1750                            <1>       ;
1751                            <1>       ; INPUT ->
1752                            <1>       ;      AL = IRQ Number
1753                            <1>       ;      EBX = Hardware Interrupt Service Address
1754                            <1>       ;
1755                            <1>       ; Note: There is not a validation check here
1756                            <1>       ;      because this procedure is called by
1757                            <1>       ;      TRDOS 386 kernel !
1758                            <1>       ;        (Even if a device driver does not exist
1759                            <1>       ;      this setting may be used by sysaudio
1760                            <1>       ;      and other system calls for hardware
1761                            <1>       ;      components which use IRQ method for I/O.)
1762                            <1>       ;
1763                            <1>       ;push  esi
1764 000134A1 0FB6F0             <1>       movzx  esi, al
1765 000134A4 66C1E602           <1>       shl    si, 2 ; * 4
1766 000134A8 899E[0A9C0100]     <1>       mov    [esi+DEV_INT_HNDLR], ebx
1767                            <1>       ;pop   esi
1768 000134AE C3                 <1>       retn
1769                            <1>
1770                            <1>
1771                            <1> sysaudio: ; AUDIO FUNCTIONS
1772                            <1>       ; 12/02/2021 (TRDOS 386 v2.0.3)
1773                            <1>       ; 28/07/2020
1774                            <1>       ; 27/07/2020
```

```
1775                    <1>     ; 10/10/2017
1776                    <1>     ; 22/06/2017
1777                    <1>     ; 28/05/2017, 04/06/2017, 05/06/2017, 10/06/2017
1778                    <1>     ; 01/05/2017, 12/05/2017, 15/05/2017, 20/05/2017
1779                    <1>     ; 21/04/2017, 22/04/2017, 23/04/2017, 24/04/2017
1780                    <1>     ; 10/04/2017, 13/04/2017, 14/04/2017, 16/04/2017
1781                    <1>     ; 03/04/2017 (VIA VT8237R)
1782                    <1>     ; 01/04/2016 (trdosk6.s -> tdosk8.s)
1783                    <1>     ; 16/05/2016 - TRDOS 386 (TRDOS v2.0)
1784                    <1>     ;
1785                    <1>     ; Inputs:
1786                    <1>     ;
1787                    <1>     ;     BH = 0 -> Beep (PC Speaker)
1788                    <1>     ;          BL = Duration Counter (1 for 1/64 second)
1789                    <1>     ;          CX = Frequency Divisor (1193180/Frequency)
1790                    <1>     ;             (1331 for 886 Hz)
1791                    <1>     ;
1792                    <1>     ;     01/04/2017
1793                    <1>     ;
1794                    <1>     ;     BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
1795                    <1>     ;          BL = 0 : PC SPEAKER
1796                    <1>     ;               1 : SOUND BLASTER 16
1797                    <1>     ;               2 : INTEL AC'97
1798                    <1>     ;               3 : VIA VT8237R (VT8233)
1799                    <1>     ;               4 : INTEL HDA
1800                    <1>     ;            5-FEh : unknown/invalid
1801                    <1>     ;              ; 04/06/2017
1802                    <1>     ;            FFh : Get current audio device id
1803                    <1>     ;
1804                    <1>     ;     BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
1805                    <1>     ;           ECX = Audio Buffer Size (must be equal to
1806                    <1>     ;                 the half of DMA buffer size)
1807                    <1>     ;           EDX = Virtual Address of the buffer
1808                    <1>     ;                 (This is not DMA buffer!)
1809                    <1>     ;
1810                    <1>     ;     BH = 3 -> INITIALIZE AUDIO DEVICE
1811                    <1>     ;          BL = 0,2 -> for Signal Response Byte
1812                    <1>     ;            CL = Signal Response Byte Value (fixed)
1813                    <1>     ;                       if BL = 0
1814                    <1>     ;                 auto increment of S.R.B. value
1815                    <1>     ;                       if BL = 2
1816                    <1>     ;              EDX = Signal Response (Return) Byte Address
1817                    <1>     ;
1818                    <1>     ;          BL = 1 for CallBack Method
1819                    <1>     ;            EDX = CallBack Service Address (Virtual)
1820                    <1>     ;
1821                    <1>     ;          BL > 2 -> invalid function
1822                    <1>     ;
1823                    <1>     ;         (Audio buffer must be allocated before
1824                    <1>     ;          initialization.)
1825                    <1>     ;
1826                    <1>     ;     BH = 4 -> START TO PLAY
1827                    <1>     ;          BL = Mode
1828                    <1>     ;              Bit 0 = mono/stereo (1 = stereo)
1829                    <1>     ;              Bit 1 = 8 bit / 16 bit (1 = 16 bit)
1830                    <1>     ;          CX = Sampling Rate (Hz)
1831                    <1>     ;
1832                    <1>     ;     BH = 5 -> PAUSE
1833                    <1>     ;          BL = Any
1834                    <1>     ;
1835                    <1>     ;     BH = 6 -> CONTINUE TO PLAY
1836                    <1>     ;          BL = Any
1837                    <1>     ;
1838                    <1>     ;     BH = 7 -> STOP
1839                    <1>     ;          BL = Any
1840                    <1>     ;
1841                    <1>     ;     BH = 8 -> RESET
1842                    <1>     ;          BL = Any
1843                    <1>     ;
1844                    <1>     ;     BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
1845                    <1>     ;          BL = Any
1846                    <1>     ;
1847                    <1>     ;     BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
1848                    <1>     ;          BL = Any
1849                    <1>     ;
1850                    <1>     ;     BH = 11 -> SET VOLUME LEVEL
1851                    <1>     ;          BL: (Bit 0 to 6)
1852                    <1>     ;              0 = Master (Playback, Lineout) volume
1853                    <1>     ;          CL = Left Channel Volume
1854                    <1>     ;          CH = Right Channel Volume
1855                    <1>     ;
1856                    <1>     ;          Note: If BL >= 80h (Bit 7 of BL is set),
1857                    <1>     ;          volume level will be set for next playing
1858                    <1>     ;          (actual volume level will not be changed
1859                    <1>     ;          immediately)
1860                    <1>     ;
1861                    <1>     ;     BH = 12 -> DISABLE AUDIO DEVICE
1862                    <1>     ;          (reset audio device and unlink dma buffer)
1863                    <1>     ;          BL = Any
1864                    <1>     ;
1865                    <1>     ;     12/05/2017
1866                    <1>     ;     BH = 13 -> MAP DMA BUFFER TO USER
1867                    <1>     ;         (for direct access to system's dma buffer)
1868                    <1>     ;
1869                    <1>     ;          ECX = map size in bytes
1870                    <1>     ;              (will be rounded up to page borders)
1871                    <1>     ;          EDX = Virtual Address of the buffer
1872                    <1>     ;              (Will be rounded up to page borders)
1873                    <1>     ;
1874                    <1>     ;     05/06/2017
1875                    <1>     ;     04/06/2017
1876                    <1>     ;     BH = 14 -> GET AUDIO DEVICE INFO
1877                    <1>     ;          BL: 0 = Audio Controller Info
1878                    <1>     ;            > 0 = Invalid for now!
1879                    <1>     ;
```

```
1880                    <1>     ;       22/06/2017
1881                    <1>     ;       BH = 15 -> GET CURRENT SOUND DATA (for graphics)
1882                    <1>     ;            BL: 0 -> PCM OUT data
1883                    <1>     ;              > 0 -> Invalid for now!
1884                    <1>     ;            ECX = 0 -> Get DMA Buffer Pointer
1885                    <1>     ;               EDX = Not Used
1886                    <1>     ;            ECX > 0 -> Byte count for buffer (EDX)
1887                    <1>     ;                EDX = Buffer Address (Virtual)
1888                    <1>     ;
1889                    <1>     ;       10/10/2017
1890                    <1>     ;       BH = 16 -> UPDATE DMA BUFFER DATA
1891                    <1>     ;                 (by using the Audio Buffer content)
1892                    <1>     ;            BL = 0 : Update dma half buffer in sequence
1893                    <1>     ;                     (automatic destination)
1894                    <1>     ;                 1 : Update 1st half of the dma buffer
1895                    <1>     ;                 2 : Update 2nd half of the dma buffer
1896                    <1>     ;                 3-FEh: Invalid!
1897                    <1>     ;                 FFh = Get current flag value
1898                    <1>     ;                      (Half buffer number -1)
1899                    <1>     ;
1900                    <1>     ;
1901                    <1>     ; Outputs:
1902                    <1>     ;
1903                    <1>     ;       For BH = 0 -> Beep
1904                    <1>     ;           None
1905                    <1>     ;
1906                    <1>     ;       01/04/2017
1907                    <1>     ;
1908                    <1>     ;       For BH = 1 -> DETECT (& ENABLE) AUDIO DEVICE
1909                    <1>     ;           AH = 0 : PC SPEAKER
1910                    <1>     ;                1 : SOUND BLASTER 16
1911                    <1>     ;                2 : INTEL AC'97
1912                    <1>     ;                3 : VIA VT8237R (VT8233)
1913                    <1>     ;                4 : INTEL HDA
1914                    <1>     ;              5-FFh : unknown/invalid
1915                    <1>     ;           AL = mode status
1916                    <1>     ;               bit 0 = mono /stereo (1 = stereo)
1917                    <1>     ;               bit 1 = 8 bit / 16 bit ( 1 = 16 bit)
1918                    <1>     ;           04/06/2017
1919                    <1>     ;           EBX = PCI DEVICE/VENDOR ID (if >0)
1920                    <1>     ;               (BX = VENDOR ID)
1921                    <1>     ;           (if CF = 1 -> Error code in EAX)
1922                    <1>     ;
1923                    <1>     ;       For BH = 2 -> ALLOCATE AUDIO BUFFER (for user)
1924                    <1>     ;           EAX = Physical Address of the buffer
1925                    <1>     ;           (if CF = 1 -> Error code in EAX)
1926                    <1>     ;
1927                    <1>     ;       For BH = 3 -> INITIALIZE AUDIO DEVICE
1928                    <1>     ;           (if CF = 1 -> Error code in EAX)
1929                    <1>     ;
1930                    <1>     ;       For BH = 4 -> START TO PLAY
1931                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1932                    <1>     ;
1933                    <1>     ;       For BH = 5 -> PAUSE
1934                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1935                    <1>     ;
1936                    <1>     ;       For BH = 6 -> CONTINUE TO PLAY
1937                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1938                    <1>     ;
1939                    <1>     ;       For BH = 7 -> STOP
1940                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1941                    <1>     ;
1942                    <1>     ;       For BH = 8 -> RESET
1943                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1944                    <1>     ;
1945                    <1>     ;       For BH = 9 -> CANCEL (CALLBACK or S.R.B. SERVICE)
1946                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1947                    <1>     ;
1948                    <1>     ;       For BH = 10 -> DEALLOCATE AUDIO BUFFER (for user)
1949                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1950                    <1>     ;
1951                    <1>     ;       For BH = 11 -> SET VOLUME LEVEL
1952                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1953                    <1>     ;
1954                    <1>     ;       For BH = 12 -> DISABLE AUDIO DEVICE
1955                    <1>     ;           none (if CF = 1 -> Error code in EAX)
1956                    <1>     ;
1957                    <1>     ;       12/05/2017
1958                    <1>     ;       For BH = 13 -> MAP DMA BUFFER TO USER
1959                    <1>     ;           EAX = Physical Address of the buffer
1960                    <1>     ;           (if CF = 1 -> Error code in EAX)
1961                    <1>     ;
1962                    <1>     ;       04/06/2017
1963                    <1>     ;       For BH = 14 -> GET AUDIO DEVICE INFO
1964                    <1>     ;       (for BL = 0) ; 05/06/2017
1965                    <1>     ;           EAX = IRQ Number in AL
1966                    <1>     ;                Audio Device Number in AH
1967                    <1>     ;           EBX = DEV/VENDOR ID
1968                    <1>     ;               (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
1969                    <1>     ;           ECX = BUS/DEV/FN
1970                    <1>     ;               (00000000BBBBBBBBDDDDDFFF00000000)
1971                    <1>     ;           EDX = NABMBAR/NAMBAR (for AC97)
1972                    <1>     ;                (Low word, DX = NAMBAR address)
1973                    <1>     ;           EDX = Base IO Addr (DX) for SB16 & VT8233
1974                    <1>     ;           (if CF = 1 -> Error code in EAX)
1975                    <1>     ;                         (ERR_DEV_NOT_RDY = 15)
1976                    <1>     ;
1977                    <1>     ;       22/06/2017
1978                    <1>     ;       For BH = 15 -> GET CURRENT SOUND DATA
1979                    <1>     ;                      (for graphics)
1980                    <1>     ;       (for BL = 0)
1981                    <1>     ;        If ECX input is 0
1982                    <1>     ;           EAX = DMA Buffer Current Position (Offset)
1983                    <1>     ;        If ECX input >  0
1984                    <1>     ;           EAX = Actual transfer count
```

```
1985                                    <1>       ;          (Sound samples will be copied from
1986                                    <1>       ;           Current DMA Buffer Position to EDX
1987                                    <1>       ;           virtual address as EAX bytes.)
1988                                    <1>       ;       ((If CF = 1 ->  Error code in EAX))
1989                                    <1>       ;
1990                                    <1>       ;
1991                                    <1>       ;      10/10/2017
1992                                    <1>       ;      For BH = 16 -> UPDATE DMA BUFFER DATA
1993                                    <1>       ;          EAX = 0, if the updated (or current)
1994                                    <1>       ;                   half buffer is DMA half buffer 1
1995                                    <1>       ;          EAX = 1, if the updated (or current)
1996                                    <1>       ;                   half buffer is DMA half buffer 2
1997                                    <1>       ;          (If CF = 1 -> Error code in EAX)
1998                                    <1>       ;
1999                                    <1>
2000 000134AF 80FF11                   <1>       cmp    bh, AUDIO1L/4
2001 000134B2 0F8354A4FFFF             <1>       jnb    sysret
2002                                    <1>
2003 000134B8 C0E702                   <1>       shl    bh, 2 ; *4
2004 000134BB 0FB6F7                   <1>       movzx  esi, bh
2005                                    <1>
2006                                    <1>       ; 22/04/2017
2007 000134BE 31C0                     <1>       xor    eax, eax
2008 000134C0 A3[64030300]             <1>       mov    [u.r0], eax  ; 0
2009                                    <1>
2010 000134C5 FF96[D0340100]           <1>       call   dword [esi+AUDIO1]
2011                                    <1>       ;jc    error
2012 000134CB E93CA4FFFF               <1>       jmp    sysret
2013                                    <1>
2014 000134D0 [E9230000]               <1> AUDIO1:      dd     _beep ; 12/02/2021
2015                                    <1>       ;dd    beep ; FUNCTION = 0 (bl = Duration Counter
2016                                    <1>                   ;                    cx = Frequency Divisor
2017 000134D4 [14350100]               <1>       dd     soundc_detect
2018 000134D8 [B0350100]               <1>       dd     sound_alloc
2019 000134DC [6E360100]               <1>       dd     soundc_init
2020 000134E0 [26380100]               <1>       dd     sound_play
2021 000134E4 [C2380100]               <1>       dd     sound_pause
2022 000134E8 [EC380100]               <1>       dd     sound_continue
2023 000134EC [16390100]               <1>       dd     sound_stop
2024 000134F0 [3F390100]               <1>       dd     soundc_reset
2025 000134F4 [70390100]               <1>       dd     soundc_cancel
2026 000134F8 [96390100]               <1>       dd     sound_dalloc
2027 000134FC [C1390100]               <1>       dd     sound_volume
2028 00013500 [133A0100]               <1>       dd     soundc_disable
2029 00013504 [853A0100]               <1>       dd     sound_dma_map
2030 00013508 [F43A0100]               <1>       dd     soundc_info
2031 0001350C [533B0100]               <1>       dd     sound_data
2032 00013510 [003C0100]               <1>       dd     sound_update
2033                                    <1>
2034                                    <1> AUDIO1L      EQU    $ - AUDIO1
2035                                    <1>
2036                                    <1> soundc_detect:
2037                                    <1>       ; FUNCTION = 1
2038                                    <1>       ; bl = Audio device type number
2039                                    <1>       ; (0= pc speaker, 1 = sound blaster 16, 2 = intel ac97
2040                                    <1>       ;  3= via vt823x, 4 = intel HDA, 0FFh= any)
2041                                    <1>
2042                                    <1>       ; 04/06/2017
2043 00013514 8A25[999C0100]           <1>       mov    ah, [audio_device]
2044 0001351A 80FBFF                   <1>       cmp    bl, 0FFh ; get current audio device id
2045 0001351D 7408                     <1>       je     short sysaudio0
2046                                    <1>
2047 0001351F 20E4                     <1>       and    ah, ah
2048 00013521 741E                     <1>       jz     short soundc_get_dev
2049                                    <1>
2050 00013523 38DC                     <1>       cmp    ah, bl
2051 00013525 7567                     <1>       jne    short soundc_dev_err
2052                                    <1>
2053                                    <1> sysaudio0:
2054 00013527 A0[9A9C0100]             <1>       mov    al, [audio_mode]
2055                                    <1> sysaudio1:
2056 0001352C A3[64030300]             <1>       mov    [u.r0], eax
2057 00013531 8B1D[A49C0100]           <1>       mov    ebx, [audio_vendor] ; (DEVICE/VENDOR ID)
2058 00013537 8B2D[60030300]           <1>       mov    ebp, [u.usp]
2059 0001353D 895D10                   <1>       mov    [ebp+16], ebx  ; ebx
2060 00013540 C3                       <1>       retn
2061                                    <1>
2062                                    <1> soundc_get_dev:
2063                                    <1>       ; 28/05/2017
2064                                    <1>       ; 03/04/2017, 24/04/2017
2065 00013541 C605[989C0100]00         <1>       mov    byte [audio_pci], 0
2066 00013548 80FB03                   <1>       cmp    bl, 3 ; VIA VT8233 (VT8237R) Audio Controller & AC97 Codec
2067                                    <1>       ;jne   short soundc_get_dev_sb
2068                                    <1>       ; 28/05/2017
2069 0001354B 7220                     <1>       jb     short soundc_get_dev_sb
2070 0001354D 773F                     <1>       ja     short soundc_dev_err  ; temporary (28/05/2017)
2071                                    <1>       ;
2072 0001354F E86D180000               <1>       call   DetectVT8233
2073 00013554 7238                     <1>       jc     short soundc_dev_err
2074                                    <1>       ; eax = 0
2075                                    <1>
2076                                    <1>       ;mov   ebx, [audio_vendor]
2077                                    <1>       ; ebx = DEVICE/VENDOR ID
2078                                    <1>       ;      DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV
2079                                    <1>
2080 00013556 B003                     <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
2081 00013558 88C4                     <1>       mov    ah, al
2082                                    <1>
2083                                    <1> soundc_get_pci_dev_ok: ; 28/05/2017
2084 0001355A FE05[989C0100]           <1>       inc    byte [audio_pci] ; = 1
2085                                    <1> soundc_get_dev_ok:
2086                                    <1>
2087                                    <1> soundc_get_dev_sb16_ok:
2088 00013560 A2[999C0100]             <1>       mov    [audio_device], al
2089 00013565 8825[9A9C0100]           <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
```

```
2090 0001356B EBBF              <1>       jmp    short sysaudio1
2091                            <1>
2092                            <1> soundc_get_dev_sb:
2093                            <1>       ; 24/04/2017
2094 0001356D 80FB01            <1>       cmp    bl, 1 ; Sound Blaster 16
2095 00013570 750E              <1>       jne    short soundc_get_dev_ich ; 28/05/2017
2096                            <1>       ;
2097 00013572 E86E1D0000        <1>       call   DetectSB
2098 00013577 7215              <1>       jc     short soundc_dev_err
2099 00013579 B801030000        <1>       mov    eax, 0301h ; Sound Blaster 16
2100 0001357E EBE0              <1>       jmp    short soundc_get_dev_sb16_ok
2101                            <1>
2102                            <1> soundc_get_dev_ich:
2103                            <1>       ; 28/05/2017
2104                            <1>       ;cmp   bl, 2 ; Intel AC'97 Audio Controller (ICH)
2105                            <1>       ;jne   short soundc_dev_err ; Temporary  (28/05/2017)
2106                            <1>       ;                           ; (Here will be modified just after
2107                            <1>       ;                           ; new sound card code will be ready!)
2108 00013580 E82F180000        <1>       call   DetectICH
2109 00013585 7207              <1>       jc     short soundc_dev_err
2110                            <1>       ;
2111 00013587 B802030000        <1>       mov    eax, 0302h ; AC'97 (ICH)
2112 0001358C EBCC              <1>       jmp    short soundc_get_pci_dev_ok
2113                            <1>
2114                            <1> soundc_dev_err:
2115 0001358E B80F000000        <1>       mov    eax, ERR_DEV_NOT_RDY ; Device not ready !
2116 00013593 EB0C              <1>       jmp    short sysaudio_err
2117                            <1>
2118                            <1> sound_buff_error:
2119 00013595 B82E000000        <1>       mov    eax, ERR_BUFFER ; Buffer error !
2120 0001359A EB05              <1>       jmp    short sysaudio_err
2121                            <1>
2122                            <1> soundc_respond_err:
2123                            <1>       ; ERR_TIME_OUT ; 'time out !' error
2124 0001359C B819000000        <1>       mov    eax, ERR_DEV_NOT_RESP ; 'device not responding !' error
2125                            <1> sysaudio_err:
2126 000135A1 A3[64030300]      <1>       mov    [u.r0], eax
2127 000135A6 A3[C8030300]      <1>       mov    [u.error], eax
2128 000135AB E93CA3FFFF        <1>       jmp    error
2129                            <1>
2130                            <1> sound_alloc:
2131                            <1>       ; FUNCTION =  2
2132                            <1>       ; ecx = audio buffer size (in bytes)
2133                            <1>       ; edx = audio buffer address (virtual)
2134                            <1>       ; 27/07/2020
2135                            <1>       ; 28/05/2017
2136                            <1>       ; 01/05/2017, 15/05/2017
2137                            <1>       ; 21/04/2017, 24/04/2017
2138 000135B0 803D[989C0100]00    <1>       cmp    byte [audio_pci], 0
2139 000135B7 7708              <1>       ja     short snd_alloc_0
2140                            <1>       ; Max. 64KB DMA buffer !!!
2141 000135B9 81F900800000      <1>       cmp    ecx, 32768
2142 000135BF 77D4              <1>       ja     short sound_buff_error
2143                            <1> snd_alloc_0:
2144                            <1>       ; 15/05/2017
2145 000135C1 81F900100000      <1>       cmp    ecx, 4096 ; PAGE_SIZE
2146 000135C7 72CC              <1>       jb     short sound_buff_error
2147                            <1>       ;
2148 000135C9 A1[AC9C0100]      <1>       mov    eax, [audio_buffer] ; audio buffer address (current)
2149 000135CE 09C0              <1>       or     eax, eax
2150 000135D0 7445              <1>       jz     short snd_alloc_2
2151                            <1>       ; audio buffer exists !
2152 000135D2 8A1D[B3030300]    <1>       mov    bl, [u.uno]
2153 000135D8 3A1D[C19C0100]    <1>       cmp    bl, [audio_user]
2154 000135DE 0F85FC000000      <1>       jne    sndc_owner_error ; not owner !
2155 000135E4 39D0              <1>       cmp    eax, edx ; same virtual buffer address ?
2156 000135E6 7508              <1>       jne    short snd_alloc_1
2157 000135E8 3B0D[B49C0100]    <1>       cmp    ecx, [audio_buff_size]
2158 000135EE 746C              <1>       je     short snd_alloc_3 ; Nothing to do !
2159                            <1>                                ; Buffer has been set already!
2160                            <1> snd_alloc_1:
2161 000135F0 51                <1>       push   ecx
2162 000135F1 52                <1>       push   edx
2163 000135F2 89C3              <1>       mov    ebx, eax ; audio buffer address (current)
2164 000135F4 8B0D[B49C0100]    <1>       mov    ecx, [audio_buff_size]
2165 000135FA E8E131FFFF        <1>       call   deallocate_user_pages
2166 000135FF 5A                <1>       pop    edx
2167 00013600 59                <1>       pop    ecx
2168 00013601 31C0              <1>       xor    eax, eax ; 0
2169 00013603 A3[AC9C0100]      <1>       mov    [audio_buffer], eax  ; 0
2170 00013608 A3[B09C0100]      <1>       mov    [audio_p_buffer], eax  ; 0
2171 0001360D A3[B49C0100]      <1>       mov    [audio_buff_size], eax
2172 00013612 A2[C19C0100]      <1>       mov    [audio_user], al ; 0
2173                            <1> snd_alloc_2:
2174 00013617 89D3              <1>       mov    ebx, edx
2175                            <1>       ; 01/05/2017
2176 00013619 BA00F0FFFF        <1>       mov    edx, ~PAGE_OFF ; truncating page offsets
2177                            <1>                           ; for aligning to page borders
2178                            <1>       ;and   eax, edx
2179 0001361E 21D3              <1>       and    ebx, edx
2180 00013620 21D1              <1>       and    ecx, edx
2181                            <1>       ; 15/05/2017
2182                            <1>       ; EAX = Beginning address (physical)
2183                            <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
2184                            <1>       ; ECX = Number of bytes to be allocated
2185 00013622 E85E2EFFFF        <1>       call   allocate_memory_block
2186 00013627 0F8268FFFFFF      <1>       jc     sound_buff_error
2187                            <1>       ; EAX = Physical address of the allocated memory block
2188                            <1>       ; ECX = Allocated bytes (as truncated to page border)
2189                            <1>       ; EBX = Virtual address (as truncated to page border)
2190 0001362D 50                <1>       push   eax
2191 0001362E 53                <1>       push   ebx
2192 0001362F 51                <1>       push   ecx
2193 00013630 E8A032FFFF        <1>       call   allocate_user_pages
2194 00013635 59                <1>       pop    ecx
```

```
2195 00013636 5B                <1>       pop    ebx
2196 00013637 58                <1>       pop    eax
2197 00013638 722A              <1>       jc     short snd_alloc_4  ; insufficient memory, buff error
2198                            <1>       ; eax = physical address of the user's audio buffer
2199                            <1>       ; ebx = virtual address of the user's audio buffer
2200                            <1>       ; ecx = buffer size (in bytes)
2201 0001363A A3[B09C0100]      <1>       mov    [audio_p_buffer], eax
2202 0001363F 891D[AC9C0100]    <1>       mov    [audio_buffer], ebx
2203 00013645 890D[B49C0100]    <1>       mov    [audio_buff_size], ecx
2204 0001364B 8A15[B3030300]    <1>       mov    dl, [u.uno]
2205 00013651 8815[C19C0100]    <1>       mov    [audio_user], dl
2206 00013657 A3[64030300]      <1>       mov    [u.r0], eax
2207                            <1> snd_alloc_3:
2208                            <1>       ; 27/07/2020
2209 0001365C C605[C09C0100]00  <1>       mov    byte [audio_flag], 0 ; clear dma half buffer flag
2210                            <1>       ;
2211 00013663 C3                <1>       retn
2212                            <1> snd_alloc_4:
2213                            <1>       ; 15/05/2017
2214                            <1>       ; EAX = Beginning address (physical)
2215                            <1>       ; ECX = Number of bytes to be deallocated
2216 00013664 E82930FFFF        <1>       call   deallocate_memory_block
2217 00013669 E927FFFFFF        <1>       jmp    sound_buff_error  ; insufficient memory, buff error
2218                            <1>
2219                            <1> soundc_init:
2220                            <1>       ; FUNCTION = 3
2221                            <1>       ; bl = method (0= s.r.b., 1= callback, 2= auto incr s.r.b.)
2222                            <1>       ; cl = signal response byte (initial or fixed) value
2223                            <1>       ; edx = signal response byte or callback address
2224                            <1>       ; 27/07/2020
2225                            <1>       ; 28/05/2017
2226                            <1>       ; 12/05/2017, 20/05/2017
2227                            <1>       ; 22/04/2017, 23/04/2017, 24/04/2017
2228                            <1>       ; 13/04/2017, 14/04/2017, 16/04/2017, 21/04/2017
2229                            <1>       ; 03/04/2017, 10/04/2017
2230                            <1>
2231 0001366E A0[999C0100]      <1>       mov    al, [audio_device]
2232 00013673 20C0              <1>       and    al, al
2233 00013675 7549              <1>       jnz    short sndc_init_6
2234                            <1>       ;
2235 00013677 C605[989C0100]00  <1>       mov    byte [audio_pci], 0
2236 0001367E 52                <1>       push   edx
2237 0001367F 53                <1>       push   ebx
2238 00013680 51                <1>       push   ecx
2239 00013681 E85F1C0000        <1>       call   DetectSB
2240 00013686 7213              <1>       jc     short sndc_init_8
2241 00013688 66B80103          <1>       mov    ax, 0301h ; Sound Blaster 16
2242 0001368C EB1E              <1>       jmp    short sndc_init_7
2243                            <1>
2244                            <1> sndc_init_11:
2245                            <1>       ; 28/05/2017
2246 0001368E E821170000        <1>       call   DetectICH ; Detect AC'97 (ICH) Audio Controller
2247 00013693 7217              <1>       jc     short sndc_init_7
2248 00013695 66B80203          <1>       mov    ax, 0302h ; Intel AC'97 Audio Device
2249 00013699 EB0B              <1>       jmp    short sndc_init_12 ; (PCI device)
2250                            <1>
2251                            <1> sndc_init_8:
2252 0001369B E821170000        <1>       call   DetectVT8233
2253                            <1>       ;jc    short sndc_init_7
2254 000136A0 72EC              <1>       jc     sndc_init_11 ; 28/05/2017
2255                            <1>       ; eax = 0
2256 000136A2 B003              <1>       mov    al, 3  ; VIA VT8237R (VT3233) Audio Controller
2257 000136A4 88C4              <1>       mov    ah, al
2258                            <1>
2259                            <1> sndc_init_12:
2260 000136A6 FE05[989C0100]    <1>       inc    byte [audio_pci] ; = 1
2261                            <1> sndc_init_7:
2262 000136AC 59                <1>       pop    ecx
2263 000136AD 5B                <1>       pop    ebx
2264 000136AE 5A                <1>       pop    edx
2265 000136AF 0F82D9FEFFFF      <1>       jc     soundc_dev_err
2266                            <1>       ;
2267 000136B5 A2[999C0100]      <1>       mov    [audio_device], al
2268 000136BA 8825[9A9C0100]    <1>       mov    [audio_mode], ah ; stereo (bit0), 16 bit (bit1) capability
2269                            <1>
2270                            <1> sndc_init_6:
2271 000136C0 833D[AC9C0100]00  <1>       cmp    dword [audio_buffer], 0
2272 000136C7 0F86C8FEFFFF      <1>       jna    sound_buff_error
2273                            <1>
2274 000136CD A0[B3030300]      <1>       mov    al, [u.uno]
2275 000136D2 8A25[C19C0100]    <1>       mov    ah, [audio_user]
2276 000136D8 08E4              <1>       or     ah, ah
2277 000136DA 7418              <1>       jz     short sndc_init0
2278 000136DC 38E0              <1>       cmp    al, ah
2279 000136DE 7419              <1>       je     short sndc_init1
2280                            <1>
2281                            <1> sndc_owner_error:
2282 000136E0 B80B000000        <1>       mov    eax, ERR_NOT_OWNER ; 'permission denied !' error
2283                            <1> sndc_perm_error:
2284 000136E5 A3[64030300]      <1>       mov    [u.r0], eax
2285 000136EA A3[C8030300]      <1>       mov    [u.error], eax
2286 000136EF E9F8A1FFFF        <1>       jmp    error
2287                            <1> sndc_init0:
2288 000136F4 A2[C19C0100]      <1>       mov    [audio_user], al
2289                            <1> sndc_init1:
2290 000136F9 8915[C49C0100]    <1>       mov    [audio_cb_addr], edx
2291 000136FF 881D[C29C0100]    <1>       mov    [audio_cb_mode], bl
2292 00013705 880D[C39C0100]    <1>       mov    [audio_srb], cl
2293                            <1>
2294                            <1>       ; 27/07/2020
2295                            <1>       ;mov   byte [audio_flag], 0  ; clear dma half buffer flag
2296                            <1>
2297                            <1>       ; 24/04/2017
2298 0001370B 803D[999C0100]03  <1>       cmp    byte [audio_device], 3 ; VT8233 (VT8237R)
2299 00013712 7438              <1>       je     short sndc_init_9
```

```
2300                                    <1>       ;ja    short soundc_respond_err ; temporary (28/05/2017)
2301 00013714 803D[999C0100]01          <1>       cmp    byte [audio_device], 1 ; SB 16
2302 0001371B 7510                      <1>       jne    short sndc_init_13
2303 0001371D BB[0A550100]              <1>       mov    ebx, sb16_int_handler
2304                                    <1>       ; Note: 'SbInit' is at 'Start to Play' stage
2305                                    <1>       ; 20/05/2017
2306 00013722 66C705[CE9C0100]08-       <1>       mov    word [audio_master_volume], 0808h ; 2/8
2306 0001372A 08                        <1>
2307 0001372B EB3F                      <1>       jmp    short sndc_init_10
2308                                    <1> sndc_init_13:
2309                                    <1>       ; 28/05/2017
2310 0001372D 803D[999C0100]02          <1>       cmp    byte [audio_device], 2 ; AC 97 (ICH)
2311 00013734 0F8562FEFFFF              <1>       jne    soundc_respond_err ; temporary (28/05/2017)
2312                                    <1>
2313 0001373A E8201F0000                <1>       call   ac97_codec_config
2314 0001373F 0F8257FEFFFF              <1>       jc     soundc_respond_err ; codec error !
2315                                    <1>
2316 00013745 BB[46580100]              <1>       mov    ebx, ac97_int_handler
2317 0001374A EB20                      <1>       jmp    short sndc_init_10
2318                                    <1>
2319                                    <1> sndc_init_9:
2320                                    <1>       ;call  reset_codec
2321                                    <1>       ;; eax = 1
2322                                    <1>       ;call  codec_io_w16 ; w32
2323 0001374C E8E7170000                <1>       call   init_codec ; 28/05/2017
2324 00013751 0F8245FEFFFF              <1>       jc     soundc_respond_err ; codec error !
2325                                    <1>
2326 00013757 E80E1A0000                <1>       call   channel_reset
2327                                    <1>
2328                                    <1>       ; setup the Codec (actually mixer registers)
2329 0001375C E82E190000                <1>         call    codec_config  ; unmute codec, set rates.
2330 00013761 0F8235FEFFFF              <1>       jc     soundc_respond_err ; codec error !
2331                                    <1>
2332 00013767 BB[FC500100]              <1>       mov    ebx, vt8233_int_handler
2333                                    <1> sndc_init_10:
2334                                    <1>       ; 13/04/2017
2335 0001376C A0[9B9C0100]              <1>       mov    al, [audio_intr] ; IRQ number
2336 00013771 E82BFDFFFF                <1>       call   set_dev_IRQ_service
2337                                    <1>
2338                                    <1>       ; SETUP (audio) INTERRUPT CALLBACK SERVICE
2339 00013776 8A1D[9B9C0100]            <1>       mov    bl, [audio_intr] ; IRQ number
2340 0001377C 8A3D[C29C0100]            <1>       mov    bh, [audio_cb_mode]
2341 00013782 FEC7                      <1>       inc    bh  ; 1 = Signal Response Byte method (fixed value)
2342                                    <1>                    ; 2 = Callback service method
2343                                    <1>                    ; 3 = Auto Increment S.R.B. method
2344 00013784 8A0D[C39C0100]            <1>       mov    cl, [audio_srb]
2345 0001378A 8B15[C49C0100]            <1>       mov    edx, [audio_cb_addr]
2346 00013790 A0[C19C0100]              <1>       mov    al, [audio_user]
2347                                    <1>       ; 14/04/2017
2348 00013795 E8E1040000                <1>       call   set_irq_callback_service
2349                                    <1>       ; 16/04/2017
2350 0001379A A3[64030300]              <1>       mov    [u.r0], eax
2351                                    <1>       ;jnc   sysret
2352 0001379F 7316                      <1>       jnc    short sndc_init2 ; 21/04/2017
2353                                    <1>       ;
2354 000137A1 A3[C8030300]              <1>       mov    dword [u.error], eax
2355                                    <1>
2356 000137A6 A0[9B9C0100]              <1>       mov    al, [audio_intr] ; IRQ number
2357 000137AB 31DB                      <1>       xor    ebx, ebx ; reset IRQ handler address
2358 000137AD E8EFFCFFFF                <1>       call   set_dev_IRQ_service
2359                                    <1>
2360 000137B2 E935A1FFFF                <1>       jmp    error
2361                                    <1>
2362                                    <1> sndc_init2:
2363                                    <1>       ; 21/04/2017
2364 000137B7 8B0D[B49C0100]            <1>       mov    ecx, [audio_buff_size] ; audio buffer size
2365 000137BD D1E1                      <1>       shl    ecx, 1 ; *2
2366 000137BF A1[B89C0100]              <1>       mov    eax, [audio_dma_buff]
2367 000137C4 21C0                      <1>       and    eax, eax
2368 000137C6 7415                      <1>       jz     short sndc_init3
2369                                    <1>
2370 000137C8 8B15[BC9C0100]            <1>       mov    edx, [audio_dmabuff_size] ; dma buffer size
2371 000137CE 39D1                      <1>       cmp    ecx, edx
2372 000137D0 744D                      <1>       je     short sndc_init5
2373                                    <1>
2374 000137D2 87CA                      <1>       xchg   ecx, edx
2375 000137D4 E8B92EFFFF                <1>       call   deallocate_memory_block
2376 000137D9 87D1                      <1>       xchg   edx, ecx
2377 000137DB 31C0                      <1>       xor    eax, eax
2378                                    <1> sndc_init3:
2379                                    <1>       ; 12/05/2017
2380 000137DD 803D[999C0100]01          <1>       cmp    byte [audio_device], 1 ; SB 16
2381 000137E4 7515                      <1>       jne    short sndc_init4
2382 000137E6 C705[B89C0100]-           <1>       mov    dword [audio_dma_buff], sb16_dma_buffer
2382 000137EC [00000200]                <1>
2383 000137F0 C705[BC9C0100]0000-       <1>       mov    dword [audio_dmabuff_size], 65536
2383 000137F8 0100                      <1>
2384                                    <1>       ;xor   eax, eax
2385                                    <1>       ;mov   [u.r0], eax ; 0 = no error, successful
2386 000137FA C3                        <1>       retn
2387                                    <1>
2388                                    <1> sndc_init4:
2389                                    <1>       ; EAX = Beginning address (physical)
2390                                    <1>       ; EAX = 0 -> Allocate mem block from the 1st proper aperture
2391                                    <1>       ; ECX = Number of bytes to be allocated (>0)
2392 000137FB E8852CFFFF                <1>       call   allocate_memory_block
2393 00013800 0F828FFDFFFF              <1>       jc     sound_buff_error
2394                                    <1>
2395                                    <1>       ; set dma buffer address and size parameters
2396 00013806 A3[B89C0100]              <1>       mov    [audio_dma_buff], eax ; dma buffer address
2397 0001380B 890D[BC9C0100]            <1>       mov    [audio_dmabuff_size], ecx ; dma buffer size
2398                                    <1> ;     ; EAX = Beginning (physical) addr of the allocated mem block
2399                                    <1> ;     ; ECX = Num of allocated bytes (rounded up to page borders)
2400                                    <1> ;     cmp    byte [audio_pci], 0 ; AC97 audio controller ?
2401                                    <1> ;     ja     short sndc_init4
```

```
                                <1> ;
                                <1> ;       ; Sound Blaster 16 uses classic DMA
                                <1> ;       mov     edx, eax
                                <1> ;       add     edx, ecx
                                <1> ;       cmp     edx, 1000000h ; 1st 16 MB
                                <1> ;       jna     short sndc_init4
                                <1> ;
                                <1> ;       ; error !
                                <1> ;       ; restore Memory Allocation Table Content
                                <1> ;       ; EAX = Beginning address (physical)
                                <1> ;       ; ECX = Number of bytes to be deallocated
                                <1> ;       call    deallocate_memory_block
                                <1> ;       ; reset dma buffer address and size parameters
                                <1> ;       xor     eax, eax ; 0
                                <1> ;       mov     [audio_dma_buff], eax ; 0
                                <1> ;       mov     [audio_dmabuff_size], ecx ; 0
                                <1> ;       jmp     sound_buff_error
                                <1> ;
                                <1> ;sndc_init4:
00013811 803D[999C0100]03       <1>         cmp     byte [audio_device], 3
                                <1>         ;jne    short sndc_init5
00013818 7506                   <1>         jne     short sndc_init14 ; 28/05/2017
0001381A E88C190000             <1>         call    set_vt8233_bdl
                                <1> sndc_init5:
                                <1>         ;sub    eax, eax ; 0
                                <1>         ;mov    [u.r0], eax ; 0 = no error, successful
0001381F C3                     <1>         retn
                                <1> sndc_init14:
00013820 E8531F0000             <1>         call    set_ac97_bdl
                                <1>         ;jmp    short sndc_init5
00013825 C3                     <1>         retn
                                <1>
                                <1> sound_play:
                                <1>         ; FUNCTION = 4
                                <1>         ; bl = Mode
                                <1>         ;       bit 0 = mono/stereo (1 = stereo)
                                <1>         ;       bit 1 = 8 bit / 16 bit (1 = 16 bit)
                                <1>         ; cx = Sampling Rate (Hz)
                                <1>
                                <1>         ; 13/06/2017
                                <1>         ; Note: Even if Mode bits are not 11b,
                                <1>         ;       AC'97 Audio Controller (&Codec)
                                <1>         ;       will play audio samples as 16 bit, stereo
                                <1>         ;       samples.
                                <1>         ;       (Program must fill the audio buffer
                                <1>         ;       as required; 8 bit samples must be converted
                                <1>         ;       to 16 bit samples and mono samples must be
                                <1>         ;       converted to stereo samples...)
                                <1>         ;
                                <1>         ; 28/07/2020
                                <1>         ; 27/07/2020
                                <1>         ; 28/05/2017
                                <1>         ; 15/05/2017, 20/05/2017
                                <1>         ; 21/04/2017, 24/04/2017
                                <1>         ; ... device check at first
00013826 A0[999C0100]           <1>         mov     al, [audio_device]
0001382B 08C0                   <1>         or      al, al ; 0 ; pc speaker or invalid
0001382D 0F84C7EBFEFF           <1>         jz      beeper_gfx ; 'video.s' ; temporary !
                                <1> ;       cmp     al, 3 ; VIA VT 8237R (vt8233)
                                <1> ;       je      short snd_play_1
                                <1> ;       cmp     al, 1 ; SB 16
                                <1> ;       jne     soundc_dev_err ; temporary !
                                <1> ;snd_play_0:
                                <1>         ; ... buffer & (buffer) owner check at second
00013833 833D[AC9C0100]00       <1>         cmp     dword [audio_buffer], 0
0001383A 0F8655FDFFFF           <1>         jna     sound_buff_error
00013840 A0[B3030300]           <1>         mov     al, [u.uno]
00013845 3A05[C19C0100]         <1>         cmp     al, [audio_user]
0001384B 0F858FFEFFFF           <1>         jne     sndc_owner_error
                                <1>
00013851 66890D[CA9C0100]       <1>         mov     [audio_freq], cx ; sample frequency (Hertz)
00013858 88D8                   <1>         mov     al, bl
0001385A 2401                   <1>         and     al, 1 ; mono/stereo (1= stereo)
0001385C FEC0                   <1>         inc     al ; channels
0001385E A2[C99C0100]           <1>         mov     [audio_stmo], al ; sound channels (1 or 2)
00013863 B008                   <1>         mov     al, 8
00013865 F6C302                 <1>         test    bl, 2 ; bits per sample (1= 16 bit)
00013868 7402                   <1>         jz      short snd_play_bps
0001386A D0E0                   <1>         shl     al, 1
                                <1> snd_play_bps:
0001386C A2[C89C0100]           <1>         mov     [audio_bps], al
                                <1>
                                <1>         ; Transfer ring 3 (user's) audio buffer content to dma buffer
00013871 8B3D[B89C0100]         <1>         mov     edi, [audio_dma_buff] ; dma buffer (ring 0)
00013877 09FF                   <1>         or      edi, edi
00013879 0F8416FDFFFF           <1>         jz      sound_buff_error
                                <1>
                                <1>         ; 27/07/2020
                                <1>
0001387F 8B35[B09C0100]         <1>         mov     esi, [audio_p_buffer] ; physical address (ring 3)
                                <1>         ;mov    ecx, [audio_buff_size] ; 15/05/2017
00013885 8B0D[BC9C0100]         <1>         mov     ecx, [audio_dmabuff_size] ; 27/07/2020
                                <1>         ;or     ecx, ecx
                                <1>         ;jz     sound_buff_error
                                <1>         ; 28/07/2020
0001388B D1E9                   <1>         shr     ecx, 1  ; dma half buffer size
                                <1>
0001388D 8035[C09C0100]01       <1>         xor     byte [audio_flag], 1 ;  0 -> 1, 1 -> 0
00013894 7502                   <1>         jnz     short snd_play_0 ; [audio_flag] = 1
                                <1>                                  ; fill dma half buffer 1
                                <1>         ; [audio_flag] = 0
                                <1>
                                <1>         ; fill dma half buffer 2
00013896 01CF                   <1>         add     edi, ecx
                                <1>
```

```
                        <1> snd_play_0:
                        <1>       ;rep   movsb
00013898 C1E902         <1>       shr    ecx, 2  ; convert byte count to dword count
0001389B F3A5           <1>       rep    movsd
                        <1>
                        <1>       ; here, if [audio_flag] = 0, interrupt handler will update
                        <1>                       ; dma half buffer 2
                        <1>                       ; (user's audio buffer data will be
                        <1>                       ; copied into dma half buffer 2)
                        <1>       ;; 20/05/2017
                        <1>       ;mov   byte [audio_flag], 1 ; next half (on next time)
                        <1>
                        <1>       ; 24/04/2017
0001389D A0[999C0100]   <1>       mov    al, [audio_device]
000138A2 3C03           <1>       cmp    al, 3 ; VT8233 (VT8237R)
000138A4 7410           <1>       je     short snd_play_1
000138A6 3C01           <1>       cmp    al, 1 ; Sound Blaster 16
000138A8 7512           <1>       jne    short snd_play_2  ; 28/05/2017
000138AA E8041B0000     <1>       call   SbInit_play
000138AF 0F82E7FCFFFF   <1>       jc     soundc_respond_err
000138B5 C3             <1>       retn
                        <1>
                        <1> snd_play_1:
000138B6 E827190000     <1>       call   vt8233_start_play
000138BB C3             <1>       retn
                        <1>
                        <1> snd_play_2:
                        <1>       ; 28/05/2017
                        <1>       ;cmp   al, 2 ; AC'97
                        <1>       ;jne   short snd_play_3
                        <1>
000138BC E8EB1E0000     <1>       call   ac97_start_play
000138C1 C3             <1>       retn
                        <1>
                        <1> ;snd_play_3:
                        <1> ;     ;call  hda_start_play
                        <1> ;     retn
                        <1>
                        <1> sound_pause:
                        <1>       ; FUNCTION = 5
                        <1>       ; Pause
                        <1>       ; 28/05/2017
                        <1>       ; 24/04/2017
                        <1>       ; 22/04/2017
000138C2 E814030000     <1>       call   snd_dev_check
000138C7 7275           <1>       jc     short snd_nothing ; temporary.
000138C9 E81A030000     <1>       call   snd_buf_check
000138CE 726E           <1>       jc     short snd_nothing ; temporary.
000138D0 A0[999C0100]   <1>       mov    al, [audio_device]
000138D5 3C03           <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
000138D7 7409           <1>       je     short snd_pause_1
000138D9 3C01           <1>       cmp    al, 1 ; Sound Blaster 16
000138DB 750A           <1>       jne    short snd_pause_2 ; 28/05/2017
000138DD E9AF1C0000     <1>       jmp    sb16_pause
                        <1> snd_pause_1:
000138E2 E9B4190000     <1>       jmp    vt8233_pause
                        <1> snd_pause_2:
                        <1>       ; 28/05/2017
                        <1>       ;cmp   al, 2 ; AC'97
                        <1>       ;jne   short snd_nothing ; temporary.
000138E7 E94E200000     <1>       jmp    ac97_pause
                        <1>
                        <1> sound_continue:
                        <1>       ; FUNCTION = 6
                        <1>       ; Continue to play
                        <1>       ; 28/05/2017
                        <1>       ; 22/04/2017
000138EC E8EA020000     <1>       call   snd_dev_check
000138F1 724B           <1>       jc     short snd_nothing ; temporary.
000138F3 E8F0020000     <1>       call   snd_buf_check
000138F8 7244           <1>       jc     short snd_nothing ; temporary.
000138FA A0[999C0100]   <1>       mov    al, [audio_device]
000138FF 3C03           <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
00013901 7409           <1>       je     short snd_cont_1
00013903 3C01           <1>       cmp    al, 1 ; Sound Blaster 16
00013905 750A           <1>       jne    short snd_cont_2 ; 28/05/2017
00013907 E9A81C0000     <1>       jmp    sb16_continue
                        <1> snd_cont_1:
0001390C E93B190000     <1>       jmp    vt8233_play
                        <1> snd_cont_2:
                        <1>       ; 28/05/2017
                        <1>       ;cmp   al, 2 ; AC'97
                        <1>       ;jne   short snd_nothing ; temporary.
00013911 E9EC1E0000     <1>       jmp    ac97_play
                        <1>
                        <1> sound_stop:
                        <1>       ; FUNCTION = 7
                        <1>       ; Stop playing
                        <1>       ; 28/05/2017
                        <1>       ; 24/05/2017
                        <1>       ; 21/04/2017, 22/04/2017, 24/04/2017
00013916 E8C0020000     <1>       call   snd_dev_check
0001391B 7221           <1>       jc     short snd_nothing ; temporary.
                        <1>       ;call  snd_buf_check
0001391D E8CF020000     <1>       call   snd_user_check ; 24/05/2017
00013922 721A           <1>       jc     short snd_nothing ; temporary.
                        <1>
00013924 A0[999C0100]   <1>       mov    al, [audio_device]
00013929 3C03           <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
0001392B 0F8471180000   <1>       je     vt8233_stop
                        <1>       ; 28/05/2017
                        <1>       ;ja    short snd_nothing
00013931 3C01           <1>       cmp    al, 1 ; Sound Blaster 16
00013933 0F849E1C0000   <1>       je     sb16_stop
                        <1>       ;cmp   al, 2
```

```
                            <1>       ;je    short ac97_stop
00013939 E9CE1F0000         <1>       jmp    ac97_stop ; temporary.
                            <1>       ;jmp   hda_stop
                            <1>
                            <1> snd_nothing:
                            <1>       ; 21/04/2017
0001393E C3                 <1>       retn
                            <1>
                            <1> soundc_reset:
                            <1>       ; FUNCTION = 8
                            <1>       ; Reset Audio Controller
                            <1>       ; 28/05/2017
                            <1>       ; 22/04/2017
0001393F E897020000         <1>       call   snd_dev_check
00013944 72F8               <1>       jc     snd_nothing ; temporary.
00013946 E89D020000         <1>       call   snd_buf_check
0001394B 72F1               <1>       jc     snd_nothing ; temporary.
                            <1>
0001394D A0[999C0100]       <1>       mov    al, [audio_device]
00013952 3C03               <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
00013954 0F844D190000       <1>       je     vt8233_reset
0001395A 77E2               <1>       ja     short snd_nothing ; temporary.
                            <1>       ;ja    hda_reset
0001395C 3C01               <1>       cmp    al, 1 ; Sound Blaster 16
0001395E 0F8527200000       <1>       jne    ac97_reset
00013964 E8C01C0000         <1>       call   sb16_reset
00013969 0F822DFCFFFF       <1>       jc     soundc_respond_err
0001396F C3                 <1>       retn
                            <1>
                            <1> soundc_cancel:
                            <1>       ; FUNCTION = 9
                            <1>       ; Cancel audio callback service
                            <1>       ; 22/04/2017
00013970 A0[C19C0100]       <1>       mov    al, [audio_user]
00013975 3A05[B3030300]     <1>       cmp    al, [u.uno]
0001397B 75C1               <1>       jne    short snd_nothing
                            <1>       ; RESET (audio) INTERRUPT CALLBACK SERVICE
0001397D 8A1D[9B9C0100]     <1>       mov    bl, [audio_intr] ; IRQ number
00013983 A0[B3030300]       <1>       mov    al, [u.uno]
00013988 28FF               <1>       sub    bh, bh ; 0 ; unlink IRQ from user service
0001398A E8EC020000         <1>       call   set_irq_callback_service
0001398F 0F8250FDFFFF       <1>       jc     sndc_perm_error ; 'permission denied' error
00013995 C3                 <1>       retn
                            <1>
                            <1> sound_dalloc:
                            <1>       ; FUNCTION = 10
                            <1>       ; Deallocate (ring 3) audio buffer
                            <1>       ; 22/04/2017
00013996 A0[C19C0100]       <1>       mov    al, [audio_user]
0001399B 3A05[B3030300]     <1>       cmp    al, [u.uno]
000139A1 759B               <1>       jne    short snd_nothing
000139A3 8B1D[AC9C0100]     <1>       mov    ebx, [audio_buffer]
                            <1>       ;or    ebx, ebx
                            <1>       ;jz    short snd_nothing
000139A9 8B0D[B49C0100]     <1>       mov    ecx, [audio_buff_size]
000139AF E82C2EFFFF         <1>       call   deallocate_user_pages
000139B4 31C0               <1>       xor    eax, eax
000139B6 A3[AC9C0100]       <1>       mov    [audio_buffer], eax ; 0
000139BB A2[C19C0100]       <1>       mov    [audio_user], al ; 0
000139C0 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>
000139C1 80FB80             <1>       cmp    bl, 80h
000139C4 720E               <1>       jb     short snd_vol_1
000139C6 0F8772FFFFFF       <1>       ja     snd_nothing ; temporary.
                            <1>       ; Set volume level for next play (BL>= 80h)
000139CC 66890D[CE9C0100]   <1>       mov    [audio_master_volume], cx
000139D3 C3                 <1>       retn
                            <1> snd_vol_1:
                            <1>       ; set volume level immediate (BL< 80h)
000139D4 80FB00             <1>       cmp    bl, 0
000139D7 0F8761FFFFFF       <1>       ja     snd_nothing ; temporary.
                            <1>
000139DD E8F9010000         <1>       call   snd_dev_check
000139E2 0F8256FFFFFF       <1>       jc     snd_nothing ; temporary.
000139E8 E8FB010000         <1>       call   snd_buf_check
000139ED 0F824BFFFFFF       <1>       jc     snd_nothing ; temporary.
                            <1>
000139F3 A0[999C0100]       <1>       mov    al, [audio_device]
000139F8 3C03               <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
000139FA 0F84C0180000       <1>       je     vt8233_volume
                            <1>       ; 28/05/2017
00013A00 0F8738FFFFFF       <1>       ja     snd_nothing ; temporary.
                            <1>       ;ja    hda_volume
                            <1>       ; Sound Blaster 16
00013A06 3C01               <1>       cmp    al, 1 ; SB 16
00013A08 0F844E1B0000       <1>       je     sb16_volume
00013A0E E90B1E0000         <1>       jmp    ac97_volume
                            <1>
                            <1> soundc_disable:
                            <1>       ; FUNCTION = 12
                            <1>       ; Disable audio device (and unlink DMA memory)
                            <1>       ; 28/05/2017
                            <1>       ; 24/05/2017
                            <1>       ; 22/04/2017
00013A13 E8C3010000         <1>       call   snd_dev_check
```

```
2717 00013A18 0F8270FBFFFF        <1>       jc     soundc_dev_err ; temporary.
2718                              <1>       ;call  snd_buf_check
2719                              <1>       ;jc    sndc_owner_error ; temporary.
2720                              <1>
2721 00013A1E A0[999C0100]        <1>       mov    al, [audio_device]
2722 00013A23 3C03                <1>       cmp    al, 3 ; VIA VT 8237R (vt8233)
2723 00013A25 7418                <1>       je     short snd_disable_1
2724 00013A27 0F8711FFFFFF        <1>       ja     snd_nothing ; temporary.
2725 00013A2D 3C01                <1>       cmp    al, 1 ; Sound Blaster 16
2726 00013A2F 7507                <1>       jne    short snd_disable_0
2727 00013A31 E8A11B0000          <1>       call   sb16_stop
2728 00013A36 EB0C                <1>       jmp    short snd_disable_2
2729                              <1> snd_disable_0:
2730 00013A38 E8CF1E0000          <1>       call   ac97_stop
2731 00013A3D EB05                <1>       jmp    short snd_disable_2
2732                              <1> snd_disable_1:
2733 00013A3F E85E170000          <1>       call   vt8233_stop
2734                              <1> snd_disable_2:
2735 00013A44 A0[9B9C0100]        <1>       mov    al, [audio_intr]
2736 00013A49 29DB                <1>       sub    ebx, ebx ; 0 = reset
2737 00013A4B E851FAFFFF          <1>       call   set_dev_IRQ_service
2738                              <1>
2739                              <1>       ;mov   al, [audio_intr]
2740 00013A50 28E4                <1>       sub    ah, ah ; 0 = reset
2741 00013A52 E8B5F6FFFF          <1>       call   set_hardware_int_vector
2742                              <1>
2743 00013A57 31C0                <1>       xor    eax, eax
2744 00013A59 A2[999C0100]        <1>       mov    byte [audio_device], al
2745 00013A5E A2[9B9C0100]        <1>       mov    byte [audio_intr], al
2746 00013A63 8705[B89C0100]      <1>       xchg   eax, [audio_dma_buff]
2747                              <1>       ; 24/05/2017
2748                              <1>       ;or    eax, eax
2749                              <1>       ;jz    short snd_disable_3
2750                              <1>       ;cmp   eax, sb16_dma_buffer ; default DMA buffer
2751                              <1>       ;je    short snd_disable_3
2752 00013A69 803D[989C0100]00    <1>       cmp    byte [audio_pci], 0 ; AC97 audio controller ?
2753 00013A70 7612                <1>       jna    short snd_disable_3
2754 00013A72 C605[989C0100]00    <1>       mov    byte [audio_pci], 0
2755                              <1>       ;sub   ecx, ecx
2756                              <1>       ;xchg  ecx, [audio_dmabuff_size]
2757 00013A79 8B0D[BC9C0100]      <1>       mov    ecx, [audio_dmabuff_size]
2758 00013A7F E80E2CFFFF          <1>       call   deallocate_memory_block
2759                              <1> snd_disable_3:
2760 00013A84 C3                  <1>       retn
2761                              <1>
2762                              <1> sound_dma_map:
2763                              <1>       ; FUNCTION = 13
2764                              <1>       ; Map audio dma buff addr to user's buffer addr
2765                              <1>       ; 12/05/2017
2766 00013A85 21C9                <1>       and    ecx, ecx
2767 00013A87 0F8408FBFFFF        <1>       jz     sound_buff_error
2768 00013A8D 803D[999C0100]01    <1>       cmp    byte [audio_device], 1
2769 00013A94 7229                <1>       jb     short snd_dma_map_1
2770                              <1> snd_dma_map_0:
2771 00013A96 A1[B89C0100]        <1>       mov    eax, [audio_dma_buff]
2772 00013A9B 21C0                <1>       and    eax, eax
2773 00013A9D 7420                <1>       jz     short snd_dma_map_1
2774                              <1>       ;
2775 00013A9F 8A1D[C19C0100]      <1>       mov    bl, [audio_user]
2776 00013AA5 08DB                <1>       or     bl, bl
2777 00013AA7 7416                <1>       jz     short snd_dma_map_1
2778 00013AA9 3A1D[B3030300]      <1>       cmp    bl, [u.uno]
2779 00013AAF 0F852BFCFFFF        <1>       jne    sndc_owner_error
2780                              <1>       ;
2781 00013AB5 8B1D[BC9C0100]      <1>       mov    ebx, [audio_dmabuff_size]
2782 00013ABB 21DB                <1>       and    ebx, ebx
2783 00013ABD 750A                <1>       jnz    short snd_dma_map_2
2784                              <1> snd_dma_map_1:
2785 00013ABF B8[00000200]        <1>       mov    eax, sb16_dma_buffer
2786 00013AC4 BB00000100          <1>       mov    ebx, 65536
2787                              <1> snd_dma_map_2:
2788 00013AC9 81C1FF0F0000        <1>       add    ecx, PAGE_SIZE-1 ; 4095
2789 00013ACF 6681E100F0          <1>       and    cx, ~PAGE_OFF ; not 4095
2790 00013AD4 39D9                <1>       cmp    ecx, ebx
2791 00013AD6 0F87B9FAFFFF        <1>       ja     sound_buff_error
2792 00013ADC 50                  <1>       push   eax
2793 00013ADD 89D3                <1>       mov    ebx, edx
2794 00013ADF C1E90C              <1>       shr    ecx, 12 ; byte count to page count
2795                              <1>       ; eax = physical address of (audio) dma buffer
2796                              <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
2797                              <1>       ; ecx = page count (>0)
2798 00013AE2 E81B2CFFFF          <1>       call   direct_memory_access
2799 00013AE7 58                  <1>       pop    eax
2800 00013AE8 0F82A7FAFFFF        <1>       jc     sound_buff_error
2801 00013AEE A3[64030300]        <1>       mov    [u.r0], eax
2802 00013AF3 C3                  <1>       retn
2803                              <1>
2804                              <1> soundc_info:
2805                              <1>       ; FUNCTION = 14
2806                              <1>       ; Get Audio Controller Info
2807                              <1>       ; 10/06/2017
2808                              <1>       ; 05/06/2017
2809 00013AF4 20DB                <1>       and    bl, bl ; 0
2810 00013AF6 740A                <1>       jz     short sndc_info_0
2811                              <1>       ; invalid parameter !
2812 00013AF8 B817000000          <1>       mov    eax, ERR_INV_PARAMETER ; 23
2813                              <1> ;sndc_inf_error:
2814                              <1> ;     mov    [u.r0], eax
2815                              <1> ;     mov    [u.error], eax
2816                              <1> ;     jmp    error
2817 00013AFD E99FFAFFFF          <1>       jmp    sysaudio_err
2818                              <1>
2819                              <1> sndc_info_0:
2820 00013B02 E8D4000000          <1>       call   snd_dev_check
2821 00013B07 0F8281FAFFFF        <1>       jc     soundc_dev_err
```

```
2822                                    <1>
2823 00013B0D 8B1D[A49C0100]            <1>       mov    ebx, [audio_vendor]
2824 00013B13 8B0D[A09C0100]            <1>       mov    ecx, [audio_dev_id]
2825                                    <1>       ;mov   al,  [audio_device]
2826 00013B19 3C02                      <1>       cmp    al, 2 ; AC'97 (ICH)
2827 00013B1B 7513                      <1>       jne    short sndc_info_1
2828                                    <1>       ; Intel AC97 (ICH) Audio Controller (=2)
2829 00013B1D 668B15[D29C0100]          <1>       mov    dx, [NABMBAR]
2830 00013B24 C1E210                    <1>       shl    edx, 16
2831 00013B27 668B15[D09C0100]          <1>       mov    dx, [NAMBAR]
2832 00013B2E EB07                      <1>       jmp    short sndc_info_2
2833                                    <1> sndc_info_1:
2834                                    <1>       ; 05/06/2017
2835                                    <1>       ; Note: Intel HDA code (here) is not ready yet!
2836                                    <1>       ; !!! SB16 or VT8233 (VT8237R) !!!
2837 00013B30 0FB715[9E9C0100]          <1>       movzx  edx, word [audio_io_base]
2838                                    <1> sndc_info_2:
2839 00013B37 88C4                      <1>       mov    ah, al ;  [audio_device]
2840 00013B39 A0[9B9C0100]              <1>       mov    al, [audio_intr]
2841                                    <1>
2842                                    <1>       ; EAX = IRQ Number in AL
2843                                    <1>       ;       Audio Device Number in AH
2844                                    <1>       ; EBX = DEV/VENDOR ID
2845                                    <1>       ;       (DDDDDDDDDDDDDDDDVVVVVVVVVVVVVVVV)
2846                                    <1>       ; ECX = BUS/DEV/FN
2847                                    <1>       ;       (00000000BBBBBBBBDDDDDFFF00000000)
2848                                    <1>       ; EDX = NABMBAR/NAMBAR (for AC97)
2849                                    <1>       ;       (Low word, DX = NAMBAR address)
2850                                    <1>       ; EDX = Base IO Addr (DX) for SB16 & VT8233
2851                                    <1>
2852                                    <1>       ; 10/06/2017
2853 00013B3E A3[64030300]              <1>       mov    [u.r0], eax
2854 00013B43 8B2D[60030300]            <1>       mov    ebp, [u.usp]
2855 00013B49 895D10                    <1>       mov    [ebp+16], ebx  ; ebx
2856 00013B4C 895514                    <1>       mov    [ebp+20], edx  ; edx
2857 00013B4F 894D18                    <1>       mov    [ebp+24], ecx  ; ecx
2858                                    <1>
2859 00013B52 C3                        <1>       retn
2860                                    <1>
2861                                    <1> sound_data:
2862                                    <1>       ; FUNCTION = 15
2863                                    <1>       ; Get Current Sound data for graphics
2864                                    <1>       ; 22/06/2017
2865                                    <1>       ;
2866 00013B53 E883000000                <1>       call   snd_dev_check
2867 00013B58 0F8230FAFFFF              <1>       jc     soundc_dev_err ; Device not ready !
2868                                    <1>
2869 00013B5E 80FB00                    <1>       cmp    bl, 0
2870 00013B61 760A                      <1>       jna    short sound_data_0
2871                                    <1>
2872                                    <1>       ; Only PCM OUT buffer data is valid for now!
2873 00013B63 B817000000                <1>       mov    eax, ERR_INV_PARAMETER  ; 23
2874 00013B68 E934FAFFFF                <1>       jmp    sysaudio_err
2875                                    <1>
2876                                    <1> sound_data_0:
2877 00013B6D A1[B89C0100]              <1>       mov    eax, [audio_dma_buff]
2878 00013B72 09C0                      <1>       or     eax, eax
2879 00013B74 0F841BFAFFFF              <1>       jz     sound_buff_error
2880                                    <1>
2881 00013B7A 803D[999C0100]04          <1>       cmp    byte [audio_device], 4 ; Intel HDA
2882 00013B81 744F                      <1>       je     short sound_data_4 ; temporary ! (22/06/2017)
2883                                    <1>
2884 00013B83 21C9                      <1>       and    ecx, ecx
2885                                    <1>       ;jnz   short sound_data_1 ; sample tranfer
2886                                    <1>
2887                                    <1>       ; Return only DMA Buffer pointer/offset...
2888                                    <1>       ; (If DMA Buffer has been mapped to user's
2889                                    <1>       ;  memory space; program can get graphics
2890                                    <1>       ;  data by using only this pointer value.)
2891                                    <1>
2892                                    <1>       ;call  get_dma_buffer_offset
2893                                    <1>       ;; eax = DMA buffer offset
2894                                    <1>       ;;     (!not half buffer offset!)
2895                                    <1>       ;mov   [u.r0], eax
2896                                    <1>       ;retn
2897                                    <1>
2898 00013B85 0F84781F0000              <1>       jz     get_dma_buffer_offset
2899                                    <1>
2900                                    <1> sound_data_1:
2901                                    <1>       ;mov   eax, [audio_dmabuff_size]
2902                                    <1>       ;shr   eax, 1 ; half buffer size
2903                                    <1>       ;cmp   ecx, eax
2904                                    <1>       ;ja    short sound_buff_error
2905                                    <1>
2906 00013B8B 3B0D[BC9C0100]            <1>       cmp    ecx, [audio_dmabuff_size]
2907 00013B91 0F87FEF9FFFF              <1>       ja     sound_buff_error
2908                                    <1>
2909 00013B97 89D0                      <1>       mov    eax, edx
2910 00013B99 25FF0F0000                <1>       and    eax, PAGE_OFF ; 4095 (0FFFh)
2911 00013B9E 81F900100000              <1>       cmp    ecx, 4096
2912 00013BA4 7605                      <1>       jna    short sound_data_2
2913 00013BA6 B900100000                <1>       mov    ecx, 4096 ; max. 1 page
2914                                    <1> sound_data_2:
2915 00013BAB 01C8                      <1>       add    eax, ecx
2916 00013BAD 3D00100000                <1>       cmp    eax, 4096
2917 00013BB2 7606                      <1>       jna    short sound_data_3
2918 00013BB4 6625FF0F                  <1>       and    ax, PAGE_OFF ; 4095 (0FFFh)
2919 00013BB8 29C1                      <1>       sub    ecx, eax
2920                                    <1>       ; here, ECX has been adjusted to fit
2921                                    <1>       ;       in page border..  (<= 4096, >0)
2922                                    <1> sound_data_3:
2923 00013BBA 51                        <1>       push   ecx
2924 00013BBB 52                        <1>       push   edx
2925 00013BBC 89D3                      <1>       mov    ebx, edx
2926 00013BBE E82D27FFFF                <1>       call   get_physical_addr
```

```
2927 00013BC3 5A                <1>       pop    edx
2928 00013BC4 59                <1>       pop    ecx
2929 00013BC5 0F82CAF9FFFF      <1>       jc     sound_buff_error
2930                            <1>
2931                            <1>       ; eax = physical address of user's buffer
2932 00013BCB 89C3              <1>       mov    ebx, eax
2933                            <1>       ; ecx = byte (transfer) count
2934                            <1>       ;call  get_current_sound_data
2935                            <1>       ;retn
2936 00013BCD E98E1E0000        <1>       jmp    get_current_sound_data
2937                            <1>
2938                            <1> sound_data_4:
2939                            <1>       ; Intel HDA code is not ready yet !
2940                            <1>       ; 22/06/2017
2941 00013BD2 31C0              <1>       xor    eax, eax
2942 00013BD4 48                <1>       dec    eax
2943 00013BD5 A3[64030300]      <1>       mov    [u.r0], eax ; 0FFFFFFFFh
2944 00013BDA C3                <1>       retn
2945                            <1>
2946                            <1> snd_dev_check:
2947                            <1>       ; 10/06/2017
2948                            <1>       ; 05/06/2017
2949                            <1>       ; 24/05/2017
2950                            <1>       ; 22/04/2017
2951                            <1>       ; 21/04/2017
2952                            <1>       ; ... device check at first
2953 00013BDB A0[999C0100]      <1>       mov    al, [audio_device]
2954 00013BE0 3C01              <1>       cmp    al, 1 ; SB 16
2955 00013BE2 7203              <1>       jb     short snd_dev_chk_retn ; error !
2956                            <1>       ;cmp   al, 4 ; Intel HDA
2957                            <1>       ;ja    short snd_dbchk_stc ; invalid !
2958                            <1>       ; 10/06/2017
2959 00013BE4 3C05              <1>       cmp    al, 5
2960 00013BE6 F5                <1>       cmc
2961                            <1> snd_dev_chk_retn:
2962 00013BE7 C3                <1>       retn
2963                            <1>
2964                            <1> snd_buf_check:
2965                            <1>       ; 10/06/2017
2966                            <1>       ; 22/04/2017
2967                            <1>       ; 21/04/2017
2968                            <1>       ; ... buffer & (buffer) owner check at second
2969 00013BE8 833D[AC9C0100]00  <1>       cmp    dword [audio_buffer], 0
2970 00013BEF 760D              <1>       jna    short snd_dbchk_stc
2971                            <1> snd_user_check:
2972 00013BF1 A0[B3030300]      <1>       mov    al, [u.uno]
2973 00013BF6 3A05[C19C0100]    <1>       cmp    al, [audio_user]
2974                            <1>       ;jne   short snd_dbchk_stc
2975                            <1>       ;retn
2976 00013BFC 74E9              <1>       je     short snd_dev_chk_retn
2977                            <1>
2978                            <1> snd_dbchk_stc:
2979 00013BFE F9                <1>       stc
2980 00013BFF C3                <1>       retn
2981                            <1>
2982                            <1> sound_update:
2983                            <1>       ; FUNCTION = 16
2984                            <1>       ; bl =
2985                            <1>       ;    0 = automatic (sequental) update (with flag switch!)
2986                            <1>       ;    1 = update dma half buffer 1 (without flag switch!)
2987                            <1>       ;    2 = update dma half buffer 2 (without flag switch!)
2988                            <1>       ;  FFh = get current flag value
2989                            <1>       ;      0 = dma half buffer 1 (will be played next)
2990                            <1>       ;      1 = dma half buffer 2 (will be played next)
2991                            <1>
2992                            <1>       ; 10/10/2017
2993                            <1>
2994                            <1>       ; ... device check at first
2995 00013C00 A0[999C0100]      <1>       mov    al, [audio_device]
2996 00013C05 08C0              <1>       or     al, al ; 0 ; pc speaker or invalid
2997 00013C07 0F8481F9FFFF      <1>       jz     soundc_dev_err
2998                            <1>
2999                            <1>       ; ... buffer & (buffer) owner check at second
3000 00013C0D 833D[AC9C0100]00  <1>       cmp    dword [audio_buffer], 0
3001 00013C14 0F867BF9FFFF      <1>       jna    sound_buff_error
3002 00013C1A A0[B3030300]      <1>       mov    al, [u.uno]
3003 00013C1F 3A05[C19C0100]    <1>       cmp    al, [audio_user]
3004 00013C25 0F85B5FAFFFF      <1>       jne    sndc_owner_error
3005                            <1>
3006                            <1>       ; Transfer ring 3 (user's) audio buffer content to dma buffer
3007 00013C2B 8B3D[B89C0100]    <1>       mov    edi, [audio_dma_buff] ; dma buffer (ring 0)
3008 00013C31 09FF              <1>       or     edi, edi
3009 00013C33 0F845CF9FFFF      <1>       jz     sound_buff_error
3010 00013C39 8B35[B09C0100]    <1>       mov    esi, [audio_p_buffer] ; physical address (ring 3)
3011 00013C3F 8B0D[B49C0100]    <1>       mov    ecx, [audio_buff_size]
3012                            <1>
3013                            <1>       ;movzx eax, byte [audio_flag]
3014 00013C45 A0[C09C0100]      <1>       mov    al, [audio_flag]
3015                            <1>
3016 00013C4A FEC3              <1>       inc    bl
3017 00013C4C 7427              <1>       jz     short snd_update_3 ; bl = 0FFh
3018 00013C4E FECB              <1>       dec    bl
3019 00013C50 7411              <1>       jz     short snd_update_0 ; bl = 0
3020                            <1>
3021 00013C52 80FB02            <1>       cmp    bl, 2
3022 00013C55 7417              <1>       je     short snd_update_1 ; dma half buffer 2
3023 00013C57 7217              <1>       jb     short snd_update_2 ; dma half buffer 1
3024                            <1>
3025                            <1>       ; invalid parameter !
3026 00013C59 B817000000        <1>       mov    eax, ERR_INV_PARAMETER ; 23
3027                            <1> ;     mov    [u.r0], eax
3028                            <1> ;     mov    [u.error], eax
3029                            <1> ;     jmp    error
3030 00013C5E E93EF9FFFF        <1>       jmp    sysaudio_err
3031                            <1>
```

```
3032                                <1> snd_update_0:
3033 00013C63 8035[C09C0100]01      <1>       xor   byte [audio_flag], 1 ; update flag !!!
3034 00013C6A 3C01                  <1>       cmp   al, 1
3035 00013C6C 7202                  <1>       jb    short snd_update_2 ; dma half buffer 1
3036                                <1> snd_update_1:
3037                                <1>       ; dma half buffer 2
3038 00013C6E 01CF                  <1>       add   edi, ecx
3039                                <1> snd_update_2:
3040                                <1>       ;rep   movsb
3041 00013C70 C1E902                <1>       shr   ecx, 2
3042 00013C73 F3A5                  <1>       rep   movsd
3043                                <1> snd_update_3:
3044 00013C75 A3[64030300]          <1>       mov   [u.r0], eax
3045                                <1>
3046 00013C7A C3                    <1>       retn
3047                                <1>
3048                                <1> set_irq_callback_service:
3049                                <1>       ; 03/08/2020
3050                                <1>       ; 10/06/2017
3051                                <1>       ; 12/05/2017
3052                                <1>       ; 24/04/2017
3053                                <1>       ; 22/04/2017
3054                                <1>       ; caller: 'syscalbac' or 'sysaudio' or ...
3055                                <1>       ; 13/04/2017, 14/04/2017, 17/04/2017
3056                                <1>       ; 24/02/2017, 26/02/2017, 28/02/2017
3057                                <1>       ; 21/02/2017 - TRDOS 386 (TRDOS v2.0)
3058                                <1>       ;
3059                                <1>       ; Link or unlink IRQ callback service to/from user (ring 3)
3060                                <1>       ;
3061                                <1>       ; INPUT ->
3062                                <1>       ;     If AL = 0, the caller is 'syscalbac';
3063                                <1>       ;        otherwise, the caller is 'sysaudio' or ...
3064                                <1>       ;        (AL = user number)
3065                                <1>       ;
3066                                <1>       ;     BL = IRQ number (Hardware interrupt request number)
3067                                <1>       ;          (0 t0 15 but IRQ 0,1,2,6,8,14,15 are prohibited)
3068                                <1>       ;          IRQ numbers 3,4,5,7,9,10,11,12,13 are valid
3069                                <1>       ;          (numbers >15 are invalid)
3070                                <1>       ;
3071                                <1>       ;     BH = 0 = Unlink IRQ (in BL) from user (ring 3) service
3072                                <1>       ;          1 = Link IRQ by using Signal Response Byte method
3073                                <1>       ;          2 = Link IRQ by using Callback service method
3074                                <1>       ;          3 = Link IRQ by using Auto Increment S.R.B. method
3075                                <1>       ;         >3 = invalid
3076                                <1>       ;            (syscallback version will return to user)
3077                                <1>       ;
3078                                <1>       ;     CL = Signal Return/Response Byte value
3079                                <1>       ;
3080                                <1>       ;     If BH = 3, kernel will put a counter value ; 03/08/2020
3081                                <1>       ;              (into the S.R.B. addr)
3082                                <1>       ;             between 0 to 255. (start value = CL+1)
3083                                <1>       ;
3084                                <1>       ;     NOTE: counter value, for example: even and odd numbers
3085                                <1>       ;           may be used for -audio- DMA buffer switch
3086                                <1>       ;           within double buffer method, etc.
3087                                <1>       ;
3088                                <1>       ;     EDX = Signal return (Response) byte address
3089                                <1>       ;                         - or -
3090                                <1>       ;           Interrupt/Callback service/routine address
3091                                <1>       ;
3092                                <1>       ;           (virtual address in user's memory space)
3093                                <1>       ;
3094                                <1>       ; OUTPUT ->
3095                                <1>       ;     CF = 0 & EAX = 0 -> Successful setting
3096                                <1>       ;     CF = 1 & EAX > 0 -> IRQ is prohibited or locked
3097                                <1>       ;                   by another process
3098                                <1>       ;             eax = ERR_PERM_DENIED -> prohibited or locked
3099                                <1>       ;             eax = ERR_INV_PARAMETER ->
3100                                <1>       ;                   invalid parameter/option or bad address
3101                                <1>       ;
3102                                <1>       ; TRDOS 386 - IRQ CALLBACK structures (parameters):
3103                                <1>       ;
3104                                <1>       ;        [u.irqlock] = 1 word, IRQ flags (0-15) that indicates
3105                                <1>       ;                  which IRQs are locked by (that) user.
3106                                <1>       ;                   Lock and unlock (by user) will change
3107                                <1>       ;                  these flags or 'terminate process' (sysexit)
3108                                <1>       ;                  will clear these flags and unlock those IRQs.
3109                                <1>       ;
3110                                <1>       ;                  Bit 0 is for IRQ 0 and Bit 15 is for IRQ 15
3111                                <1>       ;
3112                                <1>       ;        IRQ(x).owner      : 1 byte, user, [u.uno], 0 = free (unlocked)
3113                                <1>       ;
3114                                <1>       ;        IRQ(x).method : 1 byte for callback method & status
3115                                <1>       ;                     0 = Signal Response Byte method
3116                                <1>       ;                     1 = Callback service method
3117                                <1>       ;                     >1 = invalid for current 'syscalback'.
3118                                <1>       ;                  or(+) 80h = IRQ is in use by system (ring 0)
3119                                <1>       ;                              function (audio etc.) or
3120                                <1>       ;                             a device driver.
3121                                <1>       ;                  (system function will ignore the lock/owner)
3122                                <1>       ;
3123                                <1>       ;        IRQ(x).srb: 1 byte, Signal Return/Response byte value
3124                                <1>       ;                    (a fixed value by user or a counter value
3125                                <1>       ;                   from 0 to 255, which is increased by every
3126                                <1>       ;                   interrupt just before putting it into
3127                                <1>       ;                   the Signal Response byte address
3128                                <1>       ;                   (This is not used in callback serv method)
3129                                <1>       ;
3130                                <1>       ;        IRQ(x).addr      : 1 dword
3131                                <1>       ;                    Signal Response Byte address (physical)
3132                                <1>       ;                               -or-
3133                                <1>       ;                    Callback service address (virtual)
3134                                <1>       ;
3135                                <1>       ;        IRQ(x).dev: 1 byte
3136                                <1>       ;                    0 = Default device or kernel function
```

```
3137                                    <1>     ;                               -or-
3138                                    <1>     ;                     1-255 = Assigned device driver number
3139                                    <1>     ;
3140                                    <1>     ;         (x) = 3,4,5,7,9,10,11,12,13
3141                                    <1>     ;
3142                                    <1>
3143 00013C7B 80FB0F                   <1>     cmp    bl, 15
3144 00013C7E 7729                     <1>     ja     short scbs_2
3145                                    <1>
3146 00013C80 80FF03                   <1>     cmp    bh, 3
3147 00013C83 7724                     <1>     ja     short scbs_2  ; invalid parameter
3148                                    <1>
3149 00013C85 0FB6FB                   <1>     movzx  edi, bl ; save IRQ number
3150                                    <1>
3151                                    <1>            ;  IRQ 0,1,2,6,8,14,15 are prohibited
3152                                    <1>     ;IRQenum: ; 'trdosk9.s'
3153                                    <1>     ;      db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
3154                                    <1>
3155 00013C88 0FB6B7[DA490100]         <1>     movzx  esi, byte [edi+IRQenum] ; IRQ availability
3156                                    <1>                                    ; enumeration/index
3157                                    <1>     ;dec   esi
3158 00013C8F 664E                     <1>     dec    si
3159 00013C91 780F                     <1>     js     short scbs_1 ;  0 -> 0FFFFh
3160                                    <1>
3161                                    <1>     ; ESI = IRQ callback parameters index number (0 to 8)
3162                                    <1>
3163 00013C93 08FF                     <1>     or     bh, bh
3164 00013C95 7419                     <1>     jz     short scbs_4 ; unlink the IRQ (in BL)
3165                                    <1>
3166 00013C97 FECF                     <1>     dec    bh
3167                                    <1>     ; bh = method (0 = signal response byte, 1 = callback)
3168                                    <1>     ;             (2 = auto increment of signal response byte)
3169                                    <1>
3170 00013C99 80BE[4A9C0100]00         <1>     cmp    byte [esi+IRQ.owner], 0 ; locked ?
3171 00013CA0 7637                     <1>     jna    short scbs_6 ; no... OK...
3172                                    <1>
3173                                    <1> scbs_1:
3174                                    <1>     ; permission denied (prohibited IRQ)
3175 00013CA2 B80B000000               <1>     mov    eax, ERR_PERM_DENIED
3176 00013CA7 F9                       <1>     stc
3177 00013CA8 C3                       <1>     retn
3178                                    <1> scbs_2:
3179 00013CA9 F9                       <1>     stc
3180                                    <1> scbs_3:
3181 00013CAA B817000000               <1>     mov    eax, ERR_INV_PARAMETER
3182 00013CAF C3                       <1>     retn
3183                                    <1>
3184                                    <1> scbs_4: ; unlink the requested IRQ (if it belongs to current user)
3185                                    <1>     ; 10/06/2017
3186                                    <1>     ; 22/04/2017
3187                                    <1>     ; 14/04/2017
3188                                    <1>     ; If AL = 0 -> The caller is 'syscalbac'
3189 00013CB0 8AA6[4A9C0100]           <1>     mov    ah, [esi+IRQ.owner]
3190 00013CB6 3A25[B3030300]           <1>     cmp    ah, [u.uno]
3191 00013CBC 75E4                     <1>     jne    short scbs_1
3192                                    <1>
3193 00013CBE FE0D[D6030300]           <1>     dec    byte [u.irqc] ; decrease IRQ count (in use)
3194                                    <1>
3195                                    <1>     ;sub   ah, ah
3196                                    <1>     ;mov   [esi+IRQ.owner], ah ; 0 ; free !!!
3197                                    <1>     ;and   byte [esi+IRQ.method], 80h
3198                                    <1>     ;mov   [esi+IRQ.srb], ah ; 0
3199                                    <1>     ;mov   [esi+IRQ.dev], ah ; 0
3200                                    <1>     ;mov   dword [esi+IRQ.addr], 0
3201                                    <1>     ;mov   dword [u.r0], 0
3202                                    <1>
3203                                    <1>     ;mov   byte [esi+IRQ.owner], 0
3204                                    <1>
3205                                    <1>     ; 22/04/2017
3206 00013CC4 29C0                     <1>     sub    eax, eax
3207 00013CC6 8886[4A9C0100]           <1>     mov    [esi+IRQ.owner], al ; 0
3208                                    <1>     ; 10/06/2017
3209 00013CCC 8686[5C9C0100]           <1>     xchg   al, [esi+IRQ.method]
3210 00013CD2 2480                     <1>     and    al, 80h
3211 00013CD4 745E                     <1>     jz     short scbs_12
3212                                    <1>     ; Audio device must be disabled -later- ! ([IRQ.medhod] = 80h)
3213                                    <1>
3214                                    <1> ;   cmp    byte [esi+IRQ.method], 80h ; device drv or kernel extension ?
3215                                    <1> ;   jb     short scbs_12 ; bh = 0 reset to default IRQ handler
3216                                    <1> ;
3217                                    <1> ;   and    al, al
3218                                    <1> ;   jz     short  scbs_5  ; the caller is 'syscalbac'
3219                                    <1> ;   ; The caller is 'sysaudio' or ...
3220 00013CD6 30C0                     <1>     xor    al, al
3221                                    <1> ;   mov    [esi+IRQ.method], al ; 0 ; reset kernel extension flag
3222                                    <1> ;scbs_5:
3223                                    <1> ;   sub    ah, ah
3224                                    <1>     ;mov   [u.r0], eax ; 0
3225 00013CD8 C3                       <1>     retn
3226                                    <1>
3227                                    <1> scbs_6:
3228                                    <1>     ; 14/04/2017
3229 00013CD9 20C0                     <1>     and    al, al
3230 00013CDB 7405                     <1>     jz     short scbs_7  ; the caller is 'syscalbac'
3231                                    <1>     ; AL = user number ([u.uno] or [audio.user] or ...)
3232                                    <1>     ; The caller is 'sysaudio' or ...
3233                                    <1>     ;
3234                                    <1>     ; bh = method (0 = signal response byte, 1 = callback)
3235                                    <1>     ;             (2 = auto increment of signal response byte)
3236                                    <1>
3237 00013CDD 80CF80                   <1>     or     bh, 80h             ; Kernel extension flag !
3238 00013CE0 EB0A                     <1>     jmp    short scbs_8
3239                                    <1> scbs_7:
3240 00013CE2 8A86[5C9C0100]           <1>     mov    al, [esi+IRQ.method] ; >= 80h = kernel is using this IRQ
3241 00013CE8 2480                     <1>     and    al, 80h ; use only bit 7 (kernel function flag)
```

```
3242 00013CEA 08C7               <1>       or    bh, al        ; method
3243                             <1>                           ; 0 = signal response byte, 1 = callback
3244                             <1>                           ; 2 = auto increment of s.r.b.
3245                             <1> scbs_8:
3246 00013CEC A0[B3030300]       <1>       mov   al, [u.uno] ; user (process) number (1 to 16)
3247 00013CF1 8886[4A9C0100]     <1>       mov   [esi+IRQ.owner], al ; lock the IRQ for user
3248 00013CF7 88BE[5C9C0100]     <1>       mov   [esi+IRQ.method], bh
3249                             <1>
3250                             <1> ;     test  bh, 1
3251                             <1> ;     jnz   short scbs_9  ; Callback method, CX will not be used
3252                             <1> ;
3253                             <1> ;     test  bh, 2         ; use auto increment (counter) method
3254                             <1> ;     jz    short scbs_10 ; (count can be used for buffer switch)
3255                             <1> ;scbs_9:
3256                             <1> ;     xor   ecx, ecx ; 0
3257                             <1> scbs_10:
3258                             <1>       ;mov  [esi+IRQ.method], bh
3259 00013CFD 888E[659C0100]     <1>       mov   [esi+IRQ.srb], cl
3260 00013D03 C686[539C0100]00   <1>       mov   byte [esi+IRQ.dev], 0 ; device number is always 0
3261                             <1>                           ; for this system call
3262                             <1>       ;test bh, 1
3263 00013D0A 80E701             <1>       and   bh, 1 ; 17/04/2017
3264 00013D0D 7513               <1>       jnz   short scbs_11 ; callback method, use virtual address
3265                             <1>
3266 00013D0F 53                 <1>       push  ebx ; IRQ number (in BL)
3267 00013D10 89D3               <1>       mov   ebx, edx
3268                             <1>       ; ebx = virtual address
3269                             <1>       ; [u.pgdir] = page directory's physical address
3270 00013D12 FE05[EA9B0100]     <1>       inc    byte [no_page_swap] ; 1
3271                             <1>                    ; Do not add this page to swap queue
3272                             <1>                    ; and remove it from swap queue if it is
3273                             <1>                    ; on the queue.
3274 00013D18 E8D325FFFF         <1>       call  get_physical_addr
3275 00013D1D 5B                 <1>       pop   ebx
3276 00013D1E 728A               <1>       jc    scbs_3 ; invalid address !
3277                             <1>       ; eax = physical address of the virtual address in user's space
3278 00013D20 89C2               <1>       mov   edx, eax
3279                             <1> scbs_11:
3280 00013D22 66C1E602           <1>       shl   si, 2        ; byte (index) to dword (offset)
3281 00013D26 8996[6E9C0100]     <1>       mov   [esi+IRQ.addr], edx
3282                             <1>
3283 00013D2C FE05[D6030300]     <1>       inc   byte [u.irqc]; increase IRQ (in use) count
3284                             <1>
3285 00013D32 FEC7               <1>       inc    bh ; 17/04/2017
3286                             <1>       ; bh > 0 -> set to requested IRQ handler (IRQ_u_list)
3287                             <1> scbs_12:
3288 00013D34 88D8               <1>       mov   al, bl ; IRQ number
3289 00013D36 88FC               <1>       mov   ah, bh ; 0 = reset, >0 = set
3290 00013D38 E8CFF3FFFF         <1>       call  set_hardware_int_vector
3291                             <1>
3292 00013D3D 31C0               <1>       xor   eax, eax
3293                             <1>       ;mov  [u.r0], eax ; 0
3294                             <1>
3295 00013D3F C3                 <1>       retn  ; return with success (cf=0, eax=0)
3296                             <1>
3297                             <1>
3298                             <1> sysdma: ; DMA FUNCTIONS
3299                             <1>       ; 02/09/2017
3300                             <1>       ; 28/08/2017
3301                             <1>       ; 20/08/2017 - TRDOS 386 (TRDOS v2.0)
3302                             <1>       ;
3303                             <1>       ; Inputs:
3304                             <1>       ;     BH = 0 -> Allocate DMA buffer
3305                             <1>       ;        BL = 0 -> Use the system's default DMA
3306                             <1>       ;                  (SB16) Buffer
3307                             <1>       ;             Buffer Size (max.) = 65536 bytes
3308                             <1>       ;        BL > 0 -> Allocate (a new) DMA buffer
3309                             <1>       ;        ECX = DMA Buffer Size in bytes (<=128KB)
3310                             <1>       ;        EDX = Virtual Address of DMA buffer
3311                             <1>       ;
3312                             <1>       ;     BH = 1 -> Initialize (Start) DMA service
3313                             <1>       ;          BL, bit 0 to 3 = Channel Number (0 to 7)
3314                             <1>       ;          BL, bit 7 = Auto Initialized Mode
3315                             <1>       ;                  (If bit 7 is set)
3316                             <1>       ;             bit 6 = Record (read) mode (0= playback)
3317                             <1>       ;          ECX = byte count (0 = use dma buffer size)
3318                             <1>       ;          EDX = physical buffer address
3319                             <1>       ;               (0 = use dma buffer -start- address)
3320                             <1>       ;
3321                             <1>       ;     BH = 2 -> Get Current DMA Buffer Offset
3322                             <1>       ;          BL = DMA channel number
3323                             <1>       ;
3324                             <1>       ;     BH = 3 -> Get Current DMA count down value
3325                             <1>       ;          BL = DMA channel number (0 tO 7)
3326                             <1>       ;
3327                             <1>       ;     BH = 4 -> Get Current DMA channel (in progress)
3328                             <1>       ;
3329                             <1>       ;     BH = 5 -> Get System's Default DMA Buffer Address
3330                             <1>       ;
3331                             <1>       ;     BH = 6 -> Get Current DMA Buffer Address
3332                             <1>       ;
3333                             <1>       ;     BH = 7 -> Stop DMA service
3334                             <1>       ;
3335                             <1>       ;
3336                             <1>       ; Outputs:
3337                             <1>       ;
3338                             <1>       ;     For BH = 0 ; Allocate DMA buffer
3339                             <1>       ;         EAX = Physical address of DMA buffer
3340                             <1>       ;         ECX = Allocated buffer size in bytes
3341                             <1>       ;             - page count * 4096 -
3342                             <1>       ;             (may be bigger than requested)
3343                             <1>       ;         If BL input > 0,
3344                             <1>       ;            'sysalloc:' system call will be used with
3345                             <1>       ;            EBX (for 'sysalloc') = EDX (for 'sysdma')
3346                             <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>
00013D40 80FF07                         <1>       cmp   bh, 7
00013D43 7612                           <1>       jna   short sysdma_0
                                        <1>
                                        <1> sysdma_err:
00013D45 31C0                           <1>       xor   eax, eax
00013D47 48                             <1>       dec   eax ; -1
                                        <1> sysdma_perm_err:
00013D48 A3[64030300]                   <1>       mov   [u.r0], eax
00013D4D A3[C8030300]                   <1>       mov   [u.error], eax ; DMA service error !
00013D52 E9959BFFFF                     <1>       jmp   error
                                        <1>
                                        <1> sysdma_0:
00013D57 08FF                           <1>       or    bh, bh
00013D59 0F85BA000000                   <1>       jnz   sysdma_1
                                        <1>
00013D5F 20DB                           <1>       and   bl, bl
00013D61 7416                           <1>       jz    short sysdma_01
                                        <1>
                                        <1>       ; redirect system call to 'sysalloc'
00013D63 89D3                           <1>       mov   ebx, edx ; virtual address of DMA buffer
                                        <1>       ;ecx = Buffer size in bytes
                                        <1>       ; DMA buffer address <= 16MB upper limit
00013D65 BA00000001                     <1>       mov   edx, 1024*1024*16 ; 16MB limit for DMA buff
                                        <1>
00013D6A C705[DCA00100]FFFF-            <1>       mov   dword [dma_addr], 0FFFFFFFFh ; -1
00013D72 FFFF                           <1>
                                        <1>
00013D74 E9A3E5FFFF                     <1>       jmp   sysalloc
                                        <1>
                                        <1> sysdma_01:
00013D79 B8[00000200]                   <1>       mov   eax, sb16_dma_buffer
                                        <1>
00013D7E 803D[999C0100]01               <1>       cmp   byte [audio_device], 1
00013D85 722A                           <1>       jb    short sysdma_03
                                        <1>
00013D87 3B05[B89C0100]                 <1>       cmp   eax, [audio_dma_buff]
00013D8D 7507                           <1>       jne   short sysdma_02
                                        <1>
                                        <1> sysdma_0_err:
00013D8F B80B000000                     <1>       mov   eax, ERR_PERM_DENIED
00013D94 EBB2                           <1>       jmp   short sysdma_perm_err
                                        <1>
                                        <1> sysdma_02:
                                        <1>       ; Only one user is permitted for audio/dma functions
```

```
3451                                    <1>
3452 00013D96 833D[B89C0100]00          <1>       cmp   dword [audio_dma_buff], 0
3453 00013D9D 7612                      <1>       jna   short sysdma_03
3454                                    <1>
3455 00013D9F 8A1D[C19C0100]            <1>       mov   bl, [audio_user]
3456 00013DA5 08DB                      <1>       or    bl, bl
3457 00013DA7 7408                      <1>       jz    short sysdma_03
3458                                    <1>
3459 00013DA9 3A1D[B3030300]            <1>       cmp   bl, [u.uno]
3460 00013DAF 75DE                      <1>       jne   short sysdma_0_err
3461                                    <1>
3462                                    <1> sysdma_03:
3463 00013DB1 8A1D[D9A00100]            <1>       mov   bl, [dma_user]
3464 00013DB7 20DB                      <1>       and   bl, bl
3465 00013DB9 750E                      <1>       jnz   short sysdma_04
3466                                    <1>
3467 00013DBB 8A1D[B3030300]            <1>       mov   bl, [u.uno]
3468 00013DC1 881D[D9A00100]            <1>       mov   [dma_user], bl
3469                                    <1>
3470 00013DC7 EB15                      <1>       jmp   short sysdma_05
3471                                    <1>
3472                                    <1> sysdma_04:
3473 00013DC9 8B35[DCA00100]            <1>       mov   esi, [dma_addr]
3474 00013DCF 21F6                      <1>       and   esi, esi
3475 00013DD1 740B                      <1>       jz    short sysdma_05
3476                                    <1>
3477 00013DD3 46                        <1>       inc   esi ; -1 -> 0
3478 00013DD4 7408                      <1>       jz    short sysdma_05
3479                                    <1>
3480 00013DD6 3A1D[B3030300]            <1>       cmp   bl, [u.uno]
3481 00013DDC 75B1                      <1>       jne   short sysdma_0_err
3482                                    <1>
3483                                    <1> sysdma_05:
3484                                    <1>       ; edx = virtual address (user's buffer address)
3485                                    <1>       ;
3486 00013DDE 81F900000100              <1>       cmp   ecx, 65536   ; byte count (buffer size)
3487 00013DE4 0F875BFFFFFF              <1>       ja    sysdma_err
3488                                    <1>       ;
3489 00013DEA 81C1FF0F0000              <1>       add   ecx, PAGE_SIZE-1 ; 4095
3490 00013DF0 6681E100F0                <1>       and   cx, ~PAGE_OFF ; not 4095
3491                                    <1>       ;cmp  ecx, 65536
3492                                    <1>       ;ja   sysdma_err ;
3493 00013DF5 51                        <1>       push  ecx    ; buffer size (allocated pages * 4096)
3494 00013DF6 50                        <1>       push  eax    ; offset sb16_dma_buffer
3495 00013DF7 89D3                      <1>       mov   ebx, edx
3496 00013DF9 C1E90C                    <1>       shr   ecx, 12 ; byte count to page count
3497                                    <1>       ; eax = physical address of (audio) dma buffer
3498                                    <1>       ; ebx = virtual address of (audio) dma buffer (user's pgdir)
3499                                    <1>       ; ecx = page count (>0)
3500 00013DFC E80129FFFF                <1>       call  direct_memory_access
3501 00013E01 58                        <1>       pop   eax
3502 00013E02 59                        <1>       pop   ecx
3503 00013E03 0F823CFFFFFF              <1>       jc    sysdma_err
3504                                    <1>
3505 00013E09 A3[DCA00100]              <1>       mov   [dma_addr], eax
3506 00013E0E 890D[E0A00100]            <1>       mov   [dma_size], ecx ; dma buffer size (in bytes)
3507                                    <1>
3508                                    <1>       ;mov  [u.r0], eax ; DMA Buffer Address (Physical)
3509                                    <1>
3510                                    <1>       ;mov  ebp, [u.usp]  ; ebp points to user's registers
3511                                    <1>       ;mov  [ebp+24], ecx ; return to user with ecx value
3512                                    <1>
3513                                    <1>       ;jmp  sysret
3514                                    <1>
3515                                    <1>       ; 28/08/2017
3516 00013E14 E9C4000000                <1>       jmp   sysdma_51
3517                                    <1>
3518                                    <1> sysdma_1:
3519 00013E19 80FF01                    <1>       cmp   bh, 1
3520 00013E1C 0F87A6000000              <1>       ja    sysdma_5
3521                                    <1>
3522 00013E22 F6C340                    <1>       test  bl, 40h      ; record (read) mode -BL, bit 6-
3523 00013E25 0F851AFFFFFF              <1>       jnz   sysdma_err ; not ready yet!
3524                                    <1>
3525 00013E2B A1[DCA00100]              <1>       mov   eax, [dma_addr] ; physical address of dma buffer
3526 00013E30 21C0                      <1>       and   eax, eax
3527 00013E32 0F840DFFFFFF              <1>       jz    sysdma_err
3528                                    <1>
3529 00013E38 09D2                      <1>       or    edx, edx
3530 00013E3A 7504                      <1>       jnz   short sysdma_11
3531                                    <1>
3532 00013E3C 89C2                      <1>       mov   edx, eax
3533 00013E3E EB08                      <1>       jmp   short sysdma_12
3534                                    <1> sysdma_11:
3535 00013E40 39C2                      <1>       cmp   edx, eax
3536 00013E42 0F82FDFEFFFF              <1>       jb    sysdma_err
3537                                    <1> sysdma_12:
3538 00013E48 21C9                      <1>       and   ecx, ecx
3539 00013E4A 7508                      <1>       jnz   short sysdma_13
3540                                    <1>
3541 00013E4C 8B0D[E0A00100]            <1>       mov   ecx, [dma_size]
3542 00013E52 EB0C                      <1>       jmp   short sysdma_14
3543                                    <1> sysdma_13:
3544 00013E54 3B0D[E0A00100]            <1>       cmp   ecx, [dma_size]
3545 00013E5A 0F87E5FEFFFF              <1>       ja    sysdma_err
3546                                    <1> sysdma_14:
3547 00013E60 89C6                      <1>       mov   esi, eax
3548 00013E62 0335[E0A00100]            <1>       add   esi, [dma_size]
3549                                    <1>
3550 00013E68 89D0                      <1>       mov   eax, edx
3551 00013E6A 01C8                      <1>       add   eax, ecx
3552 00013E6C 0F82D3FEFFFF              <1>       jc    sysdma_err ; 02/09/2017
3553                                    <1>
3554 00013E72 39F0                      <1>       cmp   eax, esi
3555 00013E74 0F87CBFEFFFF              <1>       ja    sysdma_err
```

```
                                <1>
00013E7A 8B3D[B89C0100]         <1>       mov    edi, [audio_dma_buff]
00013E80 8B35[DCA00100]         <1>       mov    esi, [dma_addr]
                                <1>
00013E86 09FF                   <1>       or     edi, edi
00013E88 7424                   <1>       jz     short sysdma_16
                                <1>
00013E8A 803D[999C0100]01       <1>       cmp    byte [audio_device], 1
00013E91 7208                   <1>       jb     short sysdma_15
                                <1>
                                <1>       ; Sound Blaster 16
00013E93 39FE                   <1>       cmp    esi, edi
00013E95 0F84F4FEFFFF           <1>       je     sysdma_0_err ; permmission denied !
                                <1>
                                <1> sysdma_15:
00013E9B C605[DBA00100]48       <1>       mov    byte [dma_mode], 48h ; single mode playback
                                <1>
00013EA2 F6C380                 <1>       test   bl, 80h ; DMA mode - BL, bit 7, auto init -
00013EA5 7407                   <1>       jz     short sysdma_16
                                <1>       ; Auto initialized playback (write) mode
00013EA7 8005[DBA00100]10       <1>       add    byte [dma_mode], 10h ; = 58h
                                <1> sysdma_16:
00013EAE 80E307                 <1>       and    bl, 07h
00013EB1 881D[DAA00100]         <1>       mov    [dma_channel], bl
00013EB7 8915[E4A00100]         <1>       mov    [dma_start], edx
00013EBD 890D[E8A00100]         <1>       mov    [dma_count], ecx
                                <1>
                                <1>       ; 28/08/2017
                                <1>       ;call  dma_init
                                <1>       ;jmp   sysret
00013EC3 E94B010000             <1>       jmp    dma_init
                                <1>
                                <1> sysdma_5:
00013EC8 80FF05                 <1>       cmp    bh, 5
00013ECB 7223                   <1>       jb     short sysdma_3
00013ECD 0F87CE000000           <1>       ja     sysdma_6
                                <1>
                                <1>       ; Get the system's default dma buffer addr and size
00013ED3 B8[00000200]           <1>       mov    eax, sb16_dma_buffer
00013ED8 B900000100             <1>       mov    ecx, 65536 ; Buffer size in bytes
                                <1>
                                <1> sysdma_51:
                                <1>       ; 0 = there is not a dma buffer (in use or available)
00013EDD A3[64030300]           <1>       mov    [u.r0], eax
                                <1>
00013EE2 8B2D[60030300]         <1>       mov    ebp, [u.usp]  ; ebp points to user's registers
00013EE8 894D18                 <1>       mov    [ebp+24], ecx ; return to user with ecx value
                                <1>
00013EEB E91C9AFFFF             <1>       jmp    sysret
                                <1>
                                <1> sysdma_3:
00013EF0 80FF03                 <1>       cmp    bh, 3
00013EF3 7231                   <1>       jb     short sysdma_2
00013EF5 776B                   <1>       ja     short sysdma_4
                                <1>
                                <1>       ; Get current dma count down value (remain bytes)
                                <1>       ; 28/08/2017
00013EF7 0FB635[DAA00100]       <1>       movzx  esi, byte [dma_channel]
00013EFE 0FB696[124A0100]       <1>       movzx  edx, byte [dma_flip+esi]
00013F05 EE                     <1>       out    dx, al              ; flip-flop clear
00013F06 8A96[F2490100]         <1>       mov    dl, [dma_cnt+esi] ; dma count register addr
00013F0C EC                     <1>       in     al, dx
00013F0D 0FB6D8                 <1>       movzx  ebx, al
00013F10 EC                     <1>       in     al, dx
00013F11 88C7                   <1>       mov     bh, al
                                <1>
00013F13 6683FE04               <1>       cmp    si, 4 ; channel number ?
00013F17 7202                   <1>       jb     short sysdma_31 ; 8 bit dma channel
                                <1>
00013F19 D1E3                   <1>       shl    ebx, 1 ; word count to byte count
                                <1>
                                <1> sysdma_31:
00013F1B 891D[64030300]         <1>       mov    [u.r0], ebx
                                <1>
00013F21 E9E699FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysdma_2:
                                <1>       ; Get current dma buffer offset (& page)
                                <1>       ; 28/08/2017
00013F26 0FB635[DAA00100]       <1>       movzx  esi, byte [dma_channel]
00013F2D 0FB696[124A0100]       <1>       movzx  edx, byte [dma_flip+esi]
00013F34 EE                     <1>       out    dx, al              ; flip-flop clear
00013F35 8A96[EA490100]         <1>       mov    dl, [dma_adr+esi]
00013F3B EC                     <1>       in     al, dx              ; get dma position
00013F3C 0FB6D8                 <1>       movzx  ebx, al
00013F3F EC                     <1>       in     al, dx
00013F40 88C7                   <1>       mov    bh, al
                                <1>
00013F42 6683FE04               <1>       cmp    si, 4 ; channel number ?
00013F46 7202                   <1>       jb     short sysdma_21 ; 8 bit dma channel
                                <1>
00013F48 D1E3                   <1>       shl    ebx, 1 ; word offset to byte offset
                                <1>
                                <1> sysdma_21:
00013F4A 891D[64030300]         <1>       mov    [u.r0], ebx
                                <1>
00013F50 8A96[FA490100]         <1>       mov    dl, [dma_page+esi]
00013F56 EC                     <1>       in     al, dx              ; get dma page
                                <1>
                                <1>       ;add   [u.ro+2], al
00013F57 0805[66030300]         <1>       or     [u.r0+2], al
                                <1>
00013F5D E9AA99FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysdma_4:
```

```
                                <1>       ; Get current DMA channel number
                                <1>       ; 28/08/2017
00013F62 8A25[D9A00100]         <1>       mov    ah, [dma_user]
00013F68 20E4                   <1>       and    ah, ah
00013F6A 750F                   <1>       jnz    short sysdma_42
                                <1>
                                <1> sysdma_41:
                                <1>       ; Not a valid dma channel (in use)
00013F6C C705[64030300]FFFF-    <1>       mov    dword [u.r0], -1 ; 0FFFFFFFFh
00013F74 FFFF                   <1>
00013F76 E99199FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysdma_42:
00013F7B 8B35[DCA00100]         <1>       mov    esi, [dma_addr]
00013F81 21F6                   <1>       and    esi, esi
00013F83 74E7                   <1>       jz     short sysdma_41
                                <1>
00013F85 46                     <1>       inc    esi ; -1 -> 0
00013F86 74E4                   <1>       jz     short sysdma_41
                                <1>
00013F88 A0[DAA00100]           <1>       mov    al, [dma_channel]
                                <1>
00013F8D 3A25[B3030300]         <1>       cmp    ah, [u.uno]
00013F93 7502                   <1>       jne    short sysdma_43
                                <1>
00013F95 30E4                   <1>       xor    ah, ah ; DMA channel in use by current user
                                <1>
                                <1> sysdma_43:
00013F97 A3[64030300]           <1>       mov    [u.r0], eax ; AL = dma channel number
                                <1>                          ; AH > 0 if the the channel
                                <1>                          ; in use by another user/process
00013F9C E96B99FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysdma_6:
00013FA1 80FF06                 <1>       cmp    bh, 6
00013FA4 7710                   <1>       ja     short sysdma_7
                                <1>
                                <1>       ; 28/08/2017
                                <1>       ; Get current DMA buffer addr and size
00013FA6 A1[DCA00100]           <1>       mov    eax, [dma_addr] ; dma buffer address
00013FAB 8B0D[E0A00100]         <1>       mov    ecx, [dma_size] ; dma buffer size (in bytes)
                                <1>
00013FB1 E927FFFFFF             <1>       jmp    sysdma_51
                                <1>
                                <1> sysdma_7:
                                <1>       ; DMA service STOP
00013FB6 A0[B3030300]           <1>       mov    al, [u.uno]
00013FBB 3A05[D9A00100]         <1>       cmp    al, [dma_user]
00013FC1 751D                   <1>       jne    short sysdma_72
                                <1>
00013FC3 28C0                   <1>       sub    al, al ; 0
                                <1>
00013FC5 A2[D9A00100]           <1>       mov    [dma_user], al ; clear user
                                <1>
00013FCA 8605[DBA00100]         <1>       xchg   al, [dma_mode]
00013FD0 20C0                   <1>       and    al, al
                                <1>       ;jz    short sysdma_err
00013FD2 7527                   <1>       jnz    short sysdma_73
                                <1>
                                <1> sysdma_71:
00013FD4 31C0                   <1>       xor    eax, eax
00013FD6 A3[64030300]           <1>       mov    [u.r0], eax; 0
00013FDB E92C99FFFF             <1>       jmp    sysret
                                <1>
                                <1> sysdma_72:
                                <1>       ; 28/08/2017
00013FE0 803D[D9A00100]00       <1>       cmp    byte [dma_user], 0
00013FE7 76EB                   <1>       jna    short sysdma_71 ; Nothing to do !
                                <1>
00013FE9 833D[DCA00100]00       <1>       cmp    dword [dma_addr], 0
00013FF0 0F8799FDFFFF           <1>       ja     sysdma_0_err
                                <1>
00013FF6 A2[D9A00100]           <1>       mov    [dma_user], al ; reset to current user
                                <1>
                                <1> sysdma_73:
                                <1>       ; 28/08/2017
00013FFB 0FB635[DAA00100]       <1>       movzx  esi, byte [dma_channel]
00014002 0FB696[024A0100]       <1>       movzx  edx, byte [dma_mask+esi]
00014009 A0[DAA00100]           <1>       mov    al, [dma_channel]
0001400E 0C04                   <1>       or     al, 4
00014010 EE                     <1>       out    dx, al
                                <1>
00014011 EBC1                   <1>       jmp    short sysdma_71
                                <1>
                                <1> dma_init:
                                <1>       ; 28/08/2017
                                <1>       ; 20/08/2017
                                <1>       ; DMA initialization
                                <1>       ; 14/08/2017
                                <1>       ; 03/08/2017, 06/08/2017, 08/08/2017
                                <1>       ; 02/07/2017, 13/07/2017, 16/07/2017, 30/07/2017
                                <1>       ; (Derived from 'DMA_INIT' procedure in SB16MOD.ASM)
                                <1>       ; Modified for TRDOS 386 DMA buffer allocation & initialization !
                                <1>
00014013 8B1D[E4A00100]         <1>       mov    ebx, [dma_start]
00014019 8B0D[E8A00100]         <1>       mov    ecx, [dma_count]
                                <1>
0001401F 0FB635[DAA00100]       <1>       movzx  esi, byte [dma_channel]
                                <1>
00014026 6683FE04               <1>       cmp    si, 4
0001402A 7205                   <1>       jb     short gdmi1
                                <1>       ; 08/08/2017
0001402C 66D1E9                 <1>       shr    cx, 1 ; word count
0001402F D1EB                   <1>       shr    ebx, 1 ; convert byte offset to word offset
                                <1> gdmi1:
```

```
3765                                    <1>     ;mov   [dma_poff], bx ; 08/08/2017
3766 00014031 6649                      <1>     dec    cx                  ; dma size = block size - 1
3767                                    <1>
3768 00014033 0FB696[024A0100]          <1>     movzx  edx, byte [dma_mask+esi] ; 30/07/2017
3769 0001403A A0[DAA00100]              <1>     mov    al, [dma_channel]
3770 0001403F 0C04                      <1>     or     al, 4
3771 00014041 EE                        <1>     out    dx, al              ; dma channel mask
3772                                    <1>
3773 00014042 30C0                      <1>     xor    al, al ; 0 ; any value ! 08/08/2017
3774 00014044 8A96[124A0100]            <1>     mov    dl, [dma_flip+esi]
3775 0001404A EE                        <1>     out    dx, al              ; flip-flop clear
3776                                    <1>
3777 0001404B 8A96[0A4A0100]            <1>     mov    dl, [dma_mod+esi]
3778 00014051 A0[DAA00100]              <1>     mov    al, [dma_channel]  ; 13/07/2017
3779 00014056 2403                      <1>     and    al, 3
3780                                    <1>     ; 08/08/2017
3781 00014058 0A05[DBA00100]            <1>     or     al, [dma_mode] ; 58h    ; dma mode for SB16
3782 0001405E EE                        <1>     out    dx, al
3783                                    <1>
3784 0001405F 8A96[EA490100]            <1>     mov    dl, [dma_adr+esi]
3785 00014065 88D8                      <1>     mov    al, bl
3786 00014067 EE                        <1>     out    dx, al              ; offset low
3787                                    <1>
3788 00014068 88F8                      <1>     mov    al, bh
3789 0001406A EE                        <1>     out    dx, al              ; offset high
3790                                    <1>
3791 0001406B 8A96[F2490100]            <1>     mov    dl, [dma_cnt+esi]
3792 00014071 88C8                      <1>     mov    al, cl
3793 00014073 EE                        <1>     out    dx, al              ; size low
3794                                    <1>
3795 00014074 88E8                      <1>     mov    al, ch
3796 00014076 EE                        <1>     out    dx, al              ; size high
3797                                    <1>
3798 00014077 8A96[FA490100]            <1>     mov    dl, [dma_page+esi]
3799                                    <1>     ; 14/08/2017
3800 0001407D 6683FE04                  <1>     cmp    si, 4
3801 00014081 7305                      <1>     jnb    short gdmi2
3802 00014083 C1EB10                    <1>     shr    ebx, 16
3803 00014086 EB06                      <1>     jmp    short gdmi3
3804                                    <1> gdmi2:
3805                                    <1>     ; 09/08/2017
3806 00014088 C1EB0F                    <1>     shr    ebx, 15       ; complete 16 bit shift
3807 0001408B 80E3FE                    <1>     and    bl, 0FEh ; clear bit 0 (not necessary)
3808                                    <1> gdmi3:
3809 0001408E 88D8                      <1>     mov    al, bl
3810 00014090 EE                        <1>     out    dx, al              ; page
3811                                    <1>
3812 00014091 8A96[024A0100]            <1>     mov    dl, [dma_mask+esi]
3813 00014097 A0[DAA00100]              <1>     mov    al, [dma_channel]  ; 13/07/2017
3814 0001409C 2403                      <1>     and    al, 3
3815 0001409E EE                        <1>     out    dx, al              ; dma channel unmask
3816                                    <1>
3817                                    <1>     ;retn
3818                                    <1>     ; 28/08/2017
3819 0001409F E96898FFFF                <1>     jmp    sysret
3820                                    <1>
3821                                    <1> otty:
3822                                    <1> sret:
3823                                    <1> ocvt:
3824                                    <1> ctty:
3825                                    <1> cret:
3826                                    <1> ccvt:
3827                                    <1> rtty:
3828                                    <1> wtty:
3829                                    <1> rmem:
3830                                    <1> wmem:
3831                                    <1> rfd:
3832                                    <1> rhd:
3833                                    <1> wfd:
3834                                    <1> whd:
3835                                    <1> rlpt:
3836                                    <1> wlpt:
3837                                    <1> rcvt:
3838                                    <1> xmtt:
3839 000140A4 C3                        <1>       retn
3095                                        %include 'trdosk9.s' ; 04/01/2016
   1                                    <1> ; ****************************************************************************
   2                                    <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.3) - INITIALIZED DATA : trdosk9.s
   3                                    <1> ; ----------------------------------------------------------------------------
   4                                    <1> ; Last Update: 02/03/2021
   5                                    <1> ; ----------------------------------------------------------------------------
   6                                    <1> ; Beginning: 04/01/2016
   7                                    <1> ; ----------------------------------------------------------------------------
   8                                    <1> ; ----------------------------------------------------------------------------
   9                                    <1> ; Assembler: NASM version 2.15 (trdos386.s)
  10                                    <1> ; ----------------------------------------------------------------------------
  11                                    <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
  12                                    <1> ; TRDOS2.ASM (09/11/2011)
  13                                    <1> ; ****************************************************************************
  14                                    <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
  15                                    <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
  16                                    <1> ; CMD_INTR.ASM [29/01/2005] Last Update: 09/11/2011
  17                                    <1> ; FILE.ASM [29/10/2009] Last Update: 09/10/2011
  18                                    <1>
  19                                    <1> ; 12/02/2016
  20                                    <1> Last_DOS_DiskNo:
  21 000140A5 01                        <1>             db 1 ; A: = 0 & B: = 1
  22                                    <1>
  23                                    <1> Restore_CDIR:
  24 000140A6 FF                        <1>             db 0FFh ; Initial value -> any number except 0
  25                                    <1>
  26                                    <1> msg_CRLF_temp:
  27 000140A7 070D0A00                  <1>             db 07h, 0Dh, 0Ah, 0
  28                                    <1>
  29                                    <1> Magic_Bytes:
```

```
000140AB 04                  <1>             db 4
000140AC 01                  <1>             db 1
                             <1> mainprog_Version:
000140AD 07                  <1>             db 7
000140AE 5B5452444F535D204D- <1>             db "[TRDOS] Main Program v2.0.020321"
000140B7 61696E2050726F6772- <1>
000140C0 616D2076322E302E30- <1>
000140C9 3230333231          <1>
000140CE 0D0A                <1>             db 0Dh, 0Ah
000140D0 286329204572646F67- <1>             db "(c) Erdogan Tan 2005-2021"
000140D9 616E2054616E203230- <1>
000140E2 30352D32303231      <1>
000140E9 0D0A00              <1>             db 0Dh, 0Ah, 0
                             <1>
                             <1> MainProgCfgFile: ; 14/04/2016
000140EC 4D41494E50524F472E- <1>             db "MAINPROG.CFG", 0
000140F5 43464700            <1>
                             <1>
                             <1> TRDOSPromptLabel:
000140F9 5452444F53          <1>             db "TRDOS"
000140FE 00                  <1>             db 0
000140FF 00<rept>            <1>                times 5 db 0
00014104 00                  <1>             db 0
                             <1>
                             <1> ; INTERNAL COMMANDS
                             <1> Command_List:
00014105 44495200            <1> Cmd_Dir:    db "DIR", 0
00014109 434400              <1> Cmd_Cd:        db "CD", 0
0001410C 433A00              <1> Cmd_Drive:  db "C:", 0
0001410F 56455200            <1> Cmd_Ver:    db "VER", 0
00014113 4558495400          <1> Cmd_Exit:   db "EXIT", 0
00014118 50524F4D505400      <1> Cmd_Prompt: db "PROMPT", 0
0001411F 564F4C554D4500      <1> Cmd_Volume: db "VOLUME", 0
00014126 4C4F4E474E414D4500  <1> Cmd_LongName:     db "LONGNAME", 0
0001412F 4441544500          <1> Cmd_Date:   db "DATE", 0
00014134 54494D4500          <1> Cmd_Time:   db "TIME", 0
00014139 52554E00            <1> Cmd_Run:    db "RUN", 0
0001413D 53455400            <1> Cmd_Set:    db "SET", 0
00014141 434C5300            <1> Cmd_Cls:    db "CLS", 0
00014145 53484F5700          <1> Cmd_Show:   db "SHOW", 0
0001414A 44454C00            <1> Cmd_Del:    db "DEL", 0
0001414E 41545452494200      <1> Cmd_Attrib: db "ATTRIB", 0
00014155 52454E414D4500      <1> Cmd_Rename: db "RENAME", 0
0001415C 524D44495200        <1> Cmd_Rmdir:  db "RMDIR", 0
00014162 4D4B44495200        <1> Cmd_Mkdir:  db "MKDIR", 0
00014168 434F505900          <1> Cmd_Copy:   db "COPY", 0
0001416D 4D4F564500          <1> Cmd_Move:   db "MOVE", 0
00014172 5041544800          <1> Cmd_Path:   db "PATH", 0
00014177 4D454D00            <1> Cmd_Mem:    db "MEM", 0
0001417B 00                  <1>             db 0
0001417C 46494E4400          <1> Cmd_Find:   db "FIND", 0
00014181 4543484F00          <1> Cmd_Echo:   db "ECHO", 0
00014186 2A00                <1> Cmd_Remark: db "*", 0
00014188 3F00                <1> Cmd_Help:   db "?", 0
0001418A 44455649434500      <1> Cmd_Device: db "DEVICE", 0
00014191 4445564C49535400    <1> Cmd_DevList:db "DEVLIST", 0
00014199 434844495200        <1> Cmd_Chdir:  db "CHDIR", 0
0001419F 4245455000          <1> Cmd_Beep:   db "BEEP", 0
                             <1>
000141A4 00                  <1>             db 0
                             <1>
                             <1> ; 15/02/2016 (FILE.ASM, 09/10/2011)
                             <1> invalid_fname_chars:
000141A5 222728292A2B2C2F    <1>             db 22h, 27h, 28h, 29h, 2Ah, 2Bh, 2Ch, 2Fh
000141AD 3A3B3C3D3E3F40      <1>             db 3Ah, 3Bh, 3Ch, 3Dh, 3Eh, 3Fh, 40h
000141B4 5B5C5D5E60          <1>             db 5Bh, 5Ch, 5Dh, 5Eh, 60h
                             <1> sizeInvFnChars  equ ($ - invalid_fname_chars)
                             <1> ;
                             <1>
                             <1> Msg_Enter_Date:
000141B9 456E746572206E6577- <1>                 db 'Enter new date (dd-mm-yy): '
000141C2 206461746520286464- <1>
000141CB 2D6D6D2D7979293A20  <1>
000141D4 00                  <1>                 db 0
                             <1> Msg_Show_Date:
000141D5 43757272656E742064- <1>                 db   'Current date is '
000141DE 61746520697320      <1>
000141E5 30                  <1> Day:            db   '0'
000141E6 30                  <1>             db   '0'
000141E7 2F                  <1>                 db   '/'
000141E8 30                  <1> Month:          db   '0'
000141E9 30                  <1>             db   '0'
000141EA 2F                  <1>                 db   '/'
000141EB 30                  <1> Century:        db   '0'
000141EC 30                  <1>                 db   '0'
000141ED 30                  <1> Year:           db   '0'
000141EE 30                  <1>             db   '0'
000141EF 0D0A00              <1>                 db   0Dh, 0Ah, 0
                             <1>
                             <1> Msg_Enter_Time:
000141F2 456E746572206E6577- <1>             db 'Enter new time: '
000141FB 2074696D653A20      <1>
00014202 00                  <1>             db 0
                             <1> Msg_Show_Time:
00014203 43757272656E742074- <1>             db   'Current time is '
0001420C 696D6520697320      <1>
00014213 30                  <1> Hour:           db   '0'
00014214 30                  <1>             db   '0'
00014215 3A                  <1>             db   ':'
00014216 30                  <1> Minute:         db   '0'
00014217 30                  <1>             db   '0'
00014218 3A                  <1>             db   ':'
00014219 30                  <1> Second:         db   '0'
0001421A 30                  <1>             db   '0'
0001421B 0D0A00              <1>             db   0Dh, 0Ah, 0
```

```
                                <1>
                                <1> ;VolSize_Unit1:   dd 0
                                <1> ;VolSize_Unit2:   dd 0
                                <1>
                                <1> VolSize_KiloBytes:
0001421E 206B696C6F62797465-    <1>             db " kilobytes", 0Dh, 0Ah, 0
00014227 730D0A00               <1>
                                <1> VolSize_Bytes:
0001422B 2062797465730D0A00     <1>             db " bytes", 0Dh, 0Ah, 0
                                <1> Volume_in_drive:
00014234 0D0A                   <1>             db 0Dh, 0Ah
                                <1> Vol_FS_Name:
00014236 54522046533120         <1>             db "TR FS1 "
0001423D 566F6C756D6520696E-    <1>             db "Volume in drive "
00014246 20647269766520         <1>
0001424D 30                     <1> Vol_Drv_Name:   db 30h
0001424E 3A                     <1>             db ":"
0001424F 20697320               <1>             db " is "
00014253 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1> Dir_Drive_Str:
00014256 54522D444F53204472-    <1>                 db "TR-DOS Drive "
0001425F 69766520               <1>
                                <1> Dir_Drive_Name:
00014263 303A                   <1>                 db "0:"
00014265 0D0A                   <1>                 db  0Dh, 0Ah
                                <1> Vol_Str_Header:
00014267 566F6C756D65204E61-    <1>                 db "Volume Name: "
00014270 6D653A20               <1>
                                <1> Vol_Name:
00014274 00<rept>               <1>             times 64 db 0
000142B4 00                     <1>             db 0
                                <1> Vol_Serial_Header:
000142B5 0D0A                   <1>             db 0Dh, 0Ah
000142B7 566F6C756D65205365-    <1>             db "Volume Serial No: "
000142C0 7269616C204E6F3A20     <1>
                                <1> Vol_Serial1:
000142C9 30303030               <1>             db "0000"
000142CD 2D                     <1>             db "-"
                                <1> Vol_Serial2:
000142CE 30303030               <1>             db "0000"
000142D2 0D0A00                 <1>             db 0Dh, 0Ah, 0
                                <1>
                                <1> ;Vol_Tot_Sec_Str_Start:
                                <1> ;           dd 0
                                <1> Vol_Total_Sector_Header:
000142D5 0D0A                   <1>             db 0Dh, 0Ah
000142D7 566F6C756D65205369-    <1>             db "Volume Size : ", 0
000142E0 7A65203A2000           <1>
                                <1> ;Vol_Tot_Sec_Str:
                                <1> ;           db "0000000000"
                                <1> ;Vol_Tot_Sec_Str_End:
                                <1> ;           db 0
                                <1> ;Vol_Free_Sectors_Str_Start:
                                <1> ;           dd 0
                                <1> Vol_Free_Sectors_Header:
000142E6 467265652053706163-    <1>             db "Free Space  : ", 0
000142EF 6520203A2000           <1>
                                <1> ;Vol_Free_Sectors_Str:
                                <1> ;           db "0000000000"
                                <1> ;Vol_Free_Sectors_Str_End:
                                <1> ;           db 0
                                <1>
                                <1> Dir_Str_Header:
000142F5 4469726563746F7279-    <1>                 db "Directory: "
000142FE 3A20                   <1>
00014300 2F                     <1> Dir_Str_Root:   db "/"
00014301 00<rept>               <1> Dir_Str:        times 64 db 0
00014341 00000000               <1>                 dd 0
00014345 00                     <1>                 db 0
                                <1>
                                <1> Msg_Bad_Command:
00014346 42616420636F6D6D61-    <1>                 db "Bad command or file name!"
0001434F 6E64206F722066696C-    <1>
00014358 65206E616D6521         <1>
0001435F 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> msgl_drv_not_ready:
00014362 070D0A                 <1>             db 07h, 0Dh, 0Ah
                                <1>
                                <1> ; CMD_INTR.ASM - 09/11/2011 - Messages
                                <1>
                                <1> Msg_Not_Ready_Read_Err:
00014365 4472697665206E6F74-    <1>                 db "Drive not ready or read error!"
0001436E 207265616479206F72-    <1>
00014377 207265616420657272-    <1>
00014380 6F7221                 <1>
00014383 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_Not_Ready_Write_Err:
00014386 4472697665206E6F74-    <1>                 db "Drive not ready or write error!"
0001438F 207265616479206F72-    <1>
00014398 207772697465206572-    <1>
000143A1 726F7221               <1>
000143A5 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_Dir_Not_Found:
000143A8 4469726563746F7279-    <1>                 db "Directory not found!"
000143B1 206E6F7420666F756E-    <1>
000143BA 6421                   <1>
000143BC 0D0A00                 <1>                 db 0Dh, 0Ah, 0
                                <1>
                                <1> Msg_File_Not_Found:
000143BF 46696C65206E6F7420-    <1>                 db "File not found!"
000143C8 666F756E6421           <1>
000143CE 0D0A00                 <1>                 db 0Dh, 0Ah, 0
```

```
                            <1>
                            <1> Msg_File_Directory_Not_Found:
000143D1 46696C65206F722064- <1>                 db "File or directory not found!"
000143DA 69726563746F727920- <1>
000143E3 6E6F7420666F756E64- <1>
000143EC 21                 <1>
000143ED 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_LongName_Not_Found:
000143F0 4C6F6E67206E616D65- <1>                 db "Long name not found!"
000143F9 206E6F7420666F756E- <1>
00014402 6421               <1>
00014404 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> beep_Insufficient_Memory: ; 20/02/2017
00014407 0D0A               <1>                 db 0Dh, 0Ah
00014409 07                 <1>                 db 07h
                            <1> Msg_Insufficient_Memory:
0001440A 496E73756666696369- <1>                 db "Insufficient memory!"
00014413 656E74206D656D6F72- <1>
0001441C 7921               <1>
0001441E 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1>
                            <1> Msg_Error_Code:
00014421 436F6D6D616E642066- <1>                 db 'Command failed! Error code : '
0001442A 61696C656421204572- <1>
00014433 726F7220636F646520- <1>
0001443C 3A20               <1>
0001443E 303068             <1> error_code_hex: db '00h'
00014441 0A0A00             <1>                 db 0Ah, 0Ah, 0
                            <1>
                            <1> align 2
                            <1> ; 10/02/2016
                            <1> ; DIR.ASM - 09/10/2011
                            <1>
00014444 3C4449523E20202020- <1> Type_Dir:       db '<DIR>     ' ; 10 bytes
0001444D 20                 <1>
                            <1>
                            <1> File_Name:
0001444E 20<rept>           <1>                 times 12 db 20h
0001445A 20                 <1>             db 20h
                            <1> Dir_Or_FileSize:
0001445B 20<rept>           <1>                 times 10 db 20h
00014465 20                 <1>             db 20h
                            <1> File_Attribute:
00014466 20202020           <1>             dd 20202020h
0001446A 20                 <1>             db 20h
                            <1> File_Day:
0001446B 3030               <1>                 db '0','0'
0001446D 2F                 <1>             db '/'
                            <1> File_Month:
0001446E 3030               <1>                 db '0','0'
00014470 2F                 <1>             db '/'
                            <1> File_Year:
00014471 30303030           <1>                 db '0','0','0','0'
00014475 20                 <1>             db 20h
                            <1> File_Hour:
00014476 3030               <1>                 db '0','0'
00014478 3A                 <1>             db ':'
                            <1> File_Minute:
00014479 3030               <1>                 db '0','0'
0001447B 00                 <1>             db 0
                            <1>
                            <1> Decimal_File_Count_Header:
0001447C 0D0A               <1>             db 0Dh, 0Ah
                            <1> Decimal_File_Count:
0001447E 00<rept>           <1>             times 6 db 0
                            <1>
00014484 2066696C6528732920- <1> str_files:  db " file(s) & "
0001448D 2620               <1>
                            <1> Decimal_Dir_Count:
0001448F 00<rept>           <1>             times 6 db 0
                            <1> str_dirs:
00014495 206469726563746F72- <1>             db " directory(s) "
0001449E 7928732920         <1>
000144A3 0D0A00             <1>             db 0Dh, 0Ah, 0
                            <1>
000144A6 206279746528732920- <1> str_bytes:  db " byte(s) in file(s)"
000144AF 696E2066696C652873- <1>
000144B8 29                 <1>
000144B9 0D0A00             <1>             db 0Dh, 0Ah, 0
                            <1>
                            <1> ; CMD_INTR.ASM - 09/11/2011
                            <1> ; 07/10/2010
                            <1> Msg_invalid_name_chars:
000144BC 496E76616C69642066- <1>                 db "Invalid file or directory name characters!"
000144C5 696C65206F72206469- <1>
000144CE 726563746F7279206E- <1>
000144D7 616D65206368617261- <1>
000144E0 637465727321       <1>
000144E6 0D0A00             <1>             db 0Dh, 0Ah, 0
                            <1> ; 21/02/2016
000144E9 46696C65206F722064- <1> Msg_Name_Exists: db "File or directory name exists!"
000144F2 69726563746F727920- <1>
000144FB 6E616D652065786973- <1>
00014504 747321             <1>
00014507 0D0A00             <1>                 db 0Dh, 0Ah, 0
                            <1> Msg_DoYouWantMkdir:
0001450A 446F20796F75207761- <1>                 db "Do you want to make directory ", 0
00014513 6E7420746F206D616B- <1>
0001451C 65206469726563746F- <1>
00014525 72792000           <1>
00014529 2028592F4E29203F20- <1> Msg_YesNo:      db " (Y/N) ? ", 0
00014532 00                 <1>
```

```
289 00014533 000D0A00            <1> Y_N_nextline:      db 0, 0Dh, 0Ah, 0
290 00014537 4F4B2E0D0A00        <1> Msg_OK:            db "OK.", 0Dh, 0Ah, 0
291                              <1>
292                              <1> ; 27/02/2016
293                              <1> Msg_DoYouWantRmDir:
294 0001453D 446F20796F75207761- <1>                db "Do you want to delete directory ", 0
294 00014546 6E7420746F2064656C- <1>
294 0001454F 657465206469726563- <1>
294 00014558 746F72792000        <1>
295                              <1> Msg_Dir_Not_Empty:
296 0001455E 4469726563746F7279- <1>                db "Directory not empty!"
296 00014567 206E6F7420656D7074- <1>
296 00014570 7921                <1>
297 00014572 0D0A00              <1>                db 0Dh, 0Ah, 0
298                              <1>
299                              <1> Msg_DoYouWantDelete:
300 00014575 446F20796F75207761- <1>                db "Do you want to delete file ",0
300 0001457E 6E7420746F2064656C- <1>
300 00014587 6574652066696C6520- <1>
300 00014590 00                  <1>
301                              <1>
302 00014591 44656C657465642E2E- <1> Msg_Deleted:    db "Deleted...", 0Dh, 0Ah, 0
302 0001459A 2E0D0A00            <1>
303                              <1>
304                              <1> Msg_Permission_Denied:
305 0001459E 07                  <1>                db 7
306 0001459F 5065726D697373696F- <1>                db "Permission denied!", 0Dh, 0Ah, 0
306 000145A8 6E2064656E69656421- <1>
306 000145B1 0D0A00              <1>
307                              <1>
308                              <1> ; 04/03/2016
309 000145B4 4E657720            <1> Msg_New:        db "New "
310 000145B8 00                  <1>                 db 0
311                              <1> Str_Attributes:
312 000145B9 417474726962757465- <1>                db "Attributes : "
312 000145C2 73203A20            <1>
313 000145C6 4E4F524D414C        <1> Attr_Chars:     db "NORMAL"
314 000145CC 00                  <1>                 db 0
315                              <1>
316                              <1> ; 06/03/2016
317                              <1> ; CMD_INTR.ASM - 16/11/2010
318                              <1> Msg_DoYouWantRename:
319 000145CD 446F20796F75207761- <1>                db "Do you want to rename ", 0
319 000145D6 6E7420746F2072656E- <1>
319 000145DF 616D652000          <1>
320 000145E4 66696C652000        <1> Rename_File:    db "file ", 0
321 000145EA 6469726563746F7279- <1> Rename_Directory: db "directory ", 0
321 000145F3 2000                <1>
322 000145F5 00<rept>            <1> Rename_OldName: times 13 db 0
323 00014602 20617320            <1> Msg_File_rename_as: db " as "
324 00014606 00<rept>            <1> Rename_NewName: times 13 db 0
325                              <1>
326                              <1> ; 08/03/2016
327                              <1> ; CMD_INTR.ASM - 01/08/2010 - 23/04/2011
328                              <1> msg_not_same_drv:
329 00014613 4E6F742073616D6520- <1>                db "Not same drive!"
329 0001461C 647269766521        <1>
330 00014622 0D0A00              <1>                db 0Dh, 0Ah, 0
331                              <1>
332                              <1> Msg_DoYouWantMoveFile:
333 00014625 446F20796F75207761- <1>                db "Do you want to move file", 0
333 0001462E 6E7420746F206D6F76- <1>
333 00014637 652066696C6500      <1>
334                              <1>
335                              <1> msg_insufficient_disk_space:
336 0001463E 496E73756666696369- <1>                db "Insufficient disk space!"
336 00014647 656E74206469736B20- <1>
336 00014650 737061636521        <1>
337 00014656 0D0A00              <1>                db 0Dh, 0Ah, 0
338                              <1>
339                              <1> ; 01/08/2010
340                              <1> msg_source_file:
341 00014659 0D0A536F7572636520- <1>                db 0Dh, 0Ah, "Source file name      :   "
341 00014662 66696C65206E616D65- <1>
341 0001466B 2020202020203A2020- <1>
341 00014674 20                  <1>
342                              <1> msg_source_file_drv:
343 00014675 203A00              <1>                db " :", 0
344                              <1> msg_destination_file:
345 00014678 0D0A44657374696E61- <1>                db 0Dh, 0Ah, "Destination file name :   "
345 00014681 74696F6E2066696C65- <1>
345 0001468A 206E616D65203A2020- <1>
345 00014693 20                  <1>
346                              <1> msg_destination_file_drv:
347 00014694 203A00              <1>                db " :", 0
348                              <1> msg_copy_nextline:
349 00014697 0D0A00              <1>                db 0Dh, 0Ah, 0
350                              <1>
351                              <1> ; 15/03/2016
352                              <1> ; CMD_INTR.ASM
353                              <1>
354                              <1> Msg_DoYouWantOverWriteFile:
355 0001469A 446F20796F75207761- <1>                    db "Do you want to overwrite file ",0
355 000146A3 6E7420746F206F7665- <1>
355 000146AC 727772697465206669- <1>
355 000146B5 6C652000            <1>
356                              <1>
357                              <1> Msg_DoYouWantCopyFile:
358 000146B9 446F20796F75207761- <1>                    db "Do you want to copy file",0
358 000146C2 6E7420746F20636F70- <1>
358 000146CB 792066696C6500      <1>
359                              <1>
360                              <1> Msg_read_file_error_before_EOF:
361 000146D2 46696C652072656164- <1>                db "File reading error! (before EOF)"
361 000146DB 696E67206572726F72- <1>
```

```
000146E4 2120286265666F7265- <1>
000146ED 20454F4629          <1>
000146F2 0A0A00              <1>             db 0Ah, 0Ah, 0
                             <1>
                             <1> ; 18/03/2016
                             <1> ; TRDOS 386 (v2.0) mainprog copy procedure
                             <1> msg_reading:
000146F5 52656164696E672E2E- <1>             db "Reading... ", 0
000146FE 2E2000              <1>
                             <1> msg_writing:
00014701 57726974696E672E2E- <1>             db "Writing... ", 0
0001470A 2E2000              <1>
                             <1> percentagestr:
0001470D 2020202500          <1>             db "   %", 0  ; "  0%" .. "100%"
                             <1> ; 11/04/2016
                             <1> Msg_No_Set_Space:
00014712 496E73756666696369- <1>                 db "Insufficient environment space!"
0001471B 656E7420656E766972- <1>
00014724 6F6E6D656E74207370- <1>
0001472D 61636521            <1>
00014731 0D0A00              <1>                 db 0Dh, 0Ah, 0
                             <1> ; 18/04/2016
                             <1> isc_msg:
00014734 0D0A                <1>             db 0Dh, 0Ah
00014736 494E56414C49442053- <1>             db "INVALID SYSTEM CALL", 0
0001473F 595354454D2043414C- <1>
00014748 4C00                <1>
                             <1> usi_msg:
0001474A 0D0A                <1>             db 0Dh, 0Ah
0001474C 554E444546494E4544- <1>             db "UNDEFINED SOFTWARE INTERRUPT", 0
00014755 20534F465457415245- <1>
0001475E 20494E544552525550- <1>
00014767 5400                <1>
                             <1> ifc_msg:
00014769 0D0A                <1>             db 0Dh, 0Ah
0001476B 494E56414C49442046- <1>             db "INVALID FUNCTION CALL"
00014774 554E4354494F4E2043- <1>
0001477D 414C4C              <1>
                             <1> inv_msg_for_trdos_v2:
00014780 20                  <1>             db 20h
00014781 666F72205452444F53- <1>             db "for TRDOS v2!"
0001478A 20763221            <1>
0001478E 07                  <1>             db 07h
0001478F 0D0A                <1>             db 0Dh, 0Ah
00014791 0D0A                <1>             db 0Dh, 0Ah
00014793 494E5420            <1>             db "INT "
00014797 303068              <1> int_num_str: db "00h"
0001479A 0D0A                <1>             db 0Dh, 0Ah
0001479C 454158203A20        <1>             db "EAX : "
000147A2 3030303030303030- <1> eax_str:     db "00000000h", 0Dh, 0Ah
000147AB 0D0A                <1>
000147AD 454950203A20        <1>             db "EIP : "
000147B3 3030303030303030- <1> eip_str:     db "00000000h", 0Dh, 0Ah, 0
000147BC 0D0A00              <1>
                             <1>
                             <1> ; 07/10/2016
                             <1> ; Device names & parameters (for kernel devices)
                             <1>
000147BF 90                  <1> align 2
                             <1> KDEV_NAME:
000147C0 5454590000000000    <1>             db 'TTY',0,0,0,0,0 ; 1
000147C8 4D454D0000000000    <1>             db 'MEM',0,0,0,0,0 ; 2
000147D0 4644300000000000    <1>             db 'FD0',0,0,0,0,0 ; 3
000147D8 4644310000000000    <1>             db 'FD1',0,0,0,0,0 ; 4
000147E0 4844300000000000    <1>             db 'HD0',0,0,0,0,0 ; 5
000147E8 4844310000000000    <1>             db 'HD1',0,0,0,0,0 ; 6
000147F0 4844320000000000    <1>             db 'HD2',0,0,0,0,0 ; 7
000147F8 4844330000000000    <1>             db 'HD3',0,0,0,0,0 ; 8
00014800 4C505400000000000   <1>             db 'LPT',0,0,0,0,0 ; 9
00014808 5454593000000000    <1>             db 'TTY0',0,0,0,0 ; 10
00014810 5454593100000000    <1>             db 'TTY1',0,0,0,0 ; 11
00014818 5454593200000000    <1>             db 'TTY2',0,0,0,0 ; 12
00014820 5454593300000000    <1>             db 'TTY3',0,0,0,0 ; 13
00014828 5454593400000000    <1>             db 'TTY4',0,0,0,0 ; 14
00014830 5454593500000000    <1>             db 'TTY5',0,0,0,0 ; 15
00014838 5454593600000000    <1>             db 'TTY6',0,0,0,0 ; 16
00014840 5454593700000000    <1>             db 'TTY7',0,0,0,0 ; 17
00014848 5454593800000000    <1>             db 'TTY8',0,0,0,0 ; 18
00014850 5454593900000000    <1>             db 'TTY9',0,0,0,0 ; 19
00014858 434F4D3100000000    <1>             db 'COM1',0,0,0,0 ; 18
00014860 434F4D3200000000    <1>             db 'COM2',0,0,0,0 ; 19
                             <1>             ;db 'CONSOLE',0       ; 1
                             <1>             ;db 'PRINTER',0   ; 9
                             <1>             ;db 'CDROM'    ; 20
                             <1>             ;db 'CDROM0'   ; 20
                             <1>             ;db 'CDROM1'   ; 21
                             <1>             ;db 'DVD'      ; 22
                             <1>             ;db 'DVD0'     ; 22
                             <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
```

```
450                                     <1>             ;db 'SD2',0,0,0,0,0 ; 37
451                                     <1>             ;db 'SD3',0,0,0,0,0 ; 38
452                                     <1>             ;db 'SATA0'    ; 35
453                                     <1>             ;db 'SATA1'    ; 36
454                                     <1>             ;db 'SATA2'        ; 37
455                                     <1>             ;db 'SATA3'        ; 38
456                                     <1>             ;db 'PATA0',0,0,0  ; 5
457                                     <1>             ;db 'PATA1',0,0,0  ; 6
458                                     <1>             ;db 'PATA2',0,0,0  ; 7
459                                     <1>             ;db 'PATA3',0,0,0  ; 8
460                                     <1>             ;db 'WIRELESS'       ; 39
461                                     <1>             ;db 'HDMI',0,0,0,0 ; 40
462 00014868 4E554C4C00000000          <1>             db 'NULL',0,0,0,0 ; 0
463                                     <1>
464                                     <1> NumOfKernelDevNames equ ($-KDEV_NAME) / 8 ; 20 (07/10/2016)
465                                     <1>
466                                     <1> KDEV_NUMBER:
467 00014870 010203040506070809        <1>             db 1,2,3,4,5,6,7,8,9
468 00014879 0A0B0C0D0E0F101112-        <1>             db 10,11,12,13,14,15,16,17,18,19
468 00014882 13                         <1>
469 00014883 121300                     <1>             db 18,19,0
470                                     <1>
471                                     <1> NumOfKernelDevices equ $ - KDEV_NUMBER
472                                     <1>
473                                     <1> KDEV_OADDR:
474 00014886 [A4400100]                 <1>             dd otty ;tty  ; 1
475 0001488A [A4400100]                 <1>             dd sret ;mem  ; 2
476 0001488E [A4400100]                 <1>             dd sret ;fd0  ; 3
477 00014892 [A4400100]                 <1>             dd sret ;fd1  ; 4
478 00014896 [A4400100]                 <1>             dd sret ;hd0  ; 5
479 0001489A [A4400100]                 <1>             dd sret ;hd1  ; 6
480 0001489E [A4400100]                 <1>             dd sret ;hd2  ; 7
481 000148A2 [A4400100]                 <1>             dd sret ;hd3  ; 8
482 000148A6 [A4400100]                 <1>             dd sret ;lpt  ; 9
483 000148AA [A4400100]                 <1>             dd ocvt ;tty0 ; 10
484 000148AE [A4400100]                 <1>             dd ocvt ;tty1 ; 11
485 000148B2 [A4400100]                 <1>             dd ocvt ;tty2 ; 12
486 000148B6 [A4400100]                 <1>             dd ocvt ;tty3 ; 13
487 000148BA [A4400100]                 <1>             dd ocvt ;tty4 ; 14
488 000148BE [A4400100]                 <1>             dd ocvt ;tty5 ; 15
489 000148C2 [A4400100]                 <1>             dd ocvt ;tty6 ; 16
490 000148C6 [A4400100]                 <1>             dd ocvt ;tty7 ; 17
491 000148CA [A4400100]                 <1>             dd ocvt ;tty8 ; 18
492 000148CE [A4400100]                 <1>             dd ocvt ;tty9 ; 19
493                                     <1>             ;dd ocvt ;com1 ; 18
494                                     <1>             ;dd ocvt ;com2 ; 19
495 000148D2 [A4400100]                 <1>             dd sret ;null ; 20
496                                     <1> KDEV_CADDR:
497 000148D6 [A4400100]                 <1>             dd ctty ;tty  ; 1
498 000148DA [A4400100]                 <1>             dd cret ;mem  ; 2
499 000148DE [A4400100]                 <1>             dd cret ;fd0  ; 3
500 000148E2 [A4400100]                 <1>             dd cret ;fd1  ; 4
501 000148E6 [A4400100]                 <1>             dd cret ;hd0  ; 5
502 000148EA [A4400100]                 <1>             dd cret ;hd1  ; 6
503 000148EE [A4400100]                 <1>             dd cret ;hd2  ; 7
504 000148F2 [A4400100]                 <1>             dd cret ;hd3  ; 8
505 000148F6 [A4400100]                 <1>             dd cret ;lpt  ; 9
506 000148FA [A4400100]                 <1>             dd ocvt ;tty0 ; 10
507 000148FE [A4400100]                 <1>             dd ccvt ;tty1 ; 11
508 00014902 [A4400100]                 <1>             dd ccvt ;tty2 ; 12
509 00014906 [A4400100]                 <1>             dd ccvt ;tty3 ; 13
510 0001490A [A4400100]                 <1>             dd ccvt ;tty4 ; 14
511 0001490E [A4400100]                 <1>             dd ccvt ;tty5 ; 15
512 00014912 [A4400100]                 <1>             dd ccvt ;tty6 ; 16
513 00014916 [A4400100]                 <1>             dd ccvt ;tty7 ; 17
514 0001491A [A4400100]                 <1>             dd ccvt ;tty8 ; 18
515 0001491E [A4400100]                 <1>             dd ccvt ;tty9 ; 19
516                                     <1>             ;dd ccvt ;com1 ; 18
517                                     <1>             ;dd ccvt ;com2 ; 19
518 00014922 [A4400100]                 <1>             dd cret ;null ; 20
519                                     <1>
520                                     <1> KDEV_RADDR:
521 00014926 [A4400100]                 <1>             dd rtty ;tty  ; 1
522 0001492A [A4400100]                 <1>             dd rmem ;mem  ; 2
523 0001492E [A4400100]                 <1>             dd rfd  ;fd0  ; 3
524 00014932 [A4400100]                 <1>             dd rfd  ;fd1  ; 4
525 00014936 [A4400100]                 <1>             dd rhd  ;hd0  ; 5
526 0001493A [A4400100]                 <1>             dd rhd  ;hd1  ; 6
527 0001493E [A4400100]                 <1>             dd rhd  ;hd2  ; 7
528 00014942 [A4400100]                 <1>             dd rhd  ;hd3  ; 8
529 00014946 [A4400100]                 <1>             dd rlpt ;lpt  ; 9
530 0001494A [A4400100]                 <1>             dd rcvt ;tty0 ; 10
531 0001494E [A4400100]                 <1>             dd rcvt ;tty1 ; 11
532 00014952 [A4400100]                 <1>             dd rcvt ;tty2 ; 12
533 00014956 [A4400100]                 <1>             dd rcvt ;tty3 ; 13
534 0001495A [A4400100]                 <1>             dd rcvt ;tty4 ; 14
535 0001495E [A4400100]                 <1>             dd rcvt ;tty5 ; 15
536 00014962 [A4400100]                 <1>             dd rcvt ;tty6 ; 16
537 00014966 [A4400100]                 <1>             dd rcvt ;tty7 ; 17
538 0001496A [A4400100]                 <1>             dd rcvt ;tty8 ; 18
539 0001496E [A4400100]                 <1>             dd rcvt ;tty9 ; 19
540                                     <1>             ;dd rcvt ;com1 ; 18
541                                     <1>             ;dd rcvt ;com2 ; 19
542 00014972 [88340100]                 <1>             dd rnull ;null ; 20
543                                     <1> KDEV_WADDR:
544 00014976 [A4400100]                 <1>             dd wtty ;tty  ; 1
545 0001497A [A4400100]                 <1>             dd wmem ;mem  ; 2
546 0001497E [A4400100]                 <1>             dd wfd  ;fd0  ; 3
547 00014982 [A4400100]                 <1>             dd wfd  ;fd1  ; 4
548 00014986 [A4400100]                 <1>             dd whd  ;hd0  ; 5
549 0001498A [A4400100]                 <1>             dd whd  ;hd1  ; 6
550 0001498E [A4400100]                 <1>             dd whd  ;hd2  ; 7
551 00014992 [A4400100]                 <1>             dd whd  ;hd3  ; 8
552 00014996 [A4400100]                 <1>             dd wlpt ;lpt  ; 9
553 0001499A [A4400100]                 <1>             dd xmtt ;tty0 ; 10
```

```
554 0001499E [A4400100]          <1>             dd xmtt ;tty1 ; 11
555 000149A2 [A4400100]          <1>             dd xmtt ;tty2 ; 12
556 000149A6 [A4400100]          <1>             dd xmtt ;tty3 ; 13
557 000149AA [A4400100]          <1>             dd xmtt ;tty4 ; 14
558 000149AE [A4400100]          <1>             dd xmtt ;tty5 ; 15
559 000149B2 [A4400100]          <1>             dd xmtt ;tty6 ; 16
560 000149B6 [A4400100]          <1>             dd xmtt ;tty7 ; 17
561 000149BA [A4400100]          <1>             dd xmtt ;tty8 ; 18
562 000149BE [A4400100]          <1>             dd xmtt ;tty9 ; 19
563                              <1>             ;dd xmtt ;com1 ; 18
564                              <1>             ;dd xmtt ;com2 ; 19
565 000149C2 [89340100]          <1>             dd wnull ;null ; 20
566                              <1>
567                              <1> ; DEV_ACCESS bits:
568                              <1>       ; bit 0 = accessable by normal users
569                              <1>       ; bit 1 = read access permission
570                              <1>       ; bit 2 = write access permission
571                              <1>       ; bit 3 = IOCTL permission to users
572                              <1>       ; bit 4 = block device if it is set
573                              <1>       ; bit 5 = 16 bit or 1024 byte data
574                              <1>       ; bit 6 = 32 bit or 2048 byte data
575                              <1>       ; bit 7 = installable device driver
576                              <1>
577                              <1> KDEV_ACCESS: ; 08/10/2016
578 000149C6 07                  <1>             db  00000111b; tty, 1
579 000149C7 07                  <1>             db  00000111b; mem, 2
580 000149C8 8F                  <1>             db  10001111b; fd0, 3
581 000149C9 8F                  <1>             db  10001111b; fd1, 4
582 000149CA 8F                  <1>             db  10001111b; hd0, 5
583 000149CB 8F                  <1>             db  10001111b; hd1, 6
584 000149CC 8F                  <1>             db  10001111b; hd2, 7
585 000149CD 8F                  <1>             db  10001111b; hd3, 8
586 000149CE 07                  <1>             db  00000111b   ; lpt, 9
587 000149CF 07                  <1>             db  00000111b; tty0, 10
588 000149D0 07                  <1>             db  00000111b; tty1, 11
589 000149D1 07                  <1>             db  00000111b; tty2, 12
590 000149D2 07                  <1>             db  00000111b; tty3, 13
591 000149D3 07                  <1>             db  00000111b; tty4, 14
592 000149D4 07                  <1>             db  00000111b; tty5, 15
593 000149D5 07                  <1>             db  00000111b; tty6, 16
594 000149D6 07                  <1>             db  00000111b; tty7, 17
595 000149D7 07                  <1>             db  00000111b; tty8, 18
596 000149D8 07                  <1>             db  00000111b; tty9, 19
597                              <1>             ;db 00000111b; com1, 18
598                              <1>             ;db 00000111b; com2, 19
599 000149D9 00                  <1>             db  00000000b   ; null, 0
600                              <1>
601                              <1> ; 07/10/2016
602                              <1> NumOfInstallableDevices equ 8
603                              <1> NUMIDEV            equ NumOfInstallableDevices ; 8
604                              <1> NUMOFDEVICES equ NumOfKernelDevices + NumOfInstallableDevices
605                              <1>
606                              <1> ; 26/02/2017
607                              <1> ; IRQ Callback (& Signal Response Byte) service availibity
608                              <1> ; 'syscalbac'
609                              <1> ; ************************************************
610                              <1> ; IRQ 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15
611                              <1> ; --- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
612                              <1> ; --- 00 00 00 01 02 03 00 04 00 05 06 07 08 09 00 00
613                              <1> ; ************************************************
614                              <1> IRQenum:
615 000149DA 000000010203000400- <1>       db  0,0,0,1,2,3,0,4,0,5,6,7,8,9,0,0
615 000149E3 05060708090000      <1>
616                              <1>
617                              <1> ; 28/08/2017
618                              <1> ; 20/08/2017
619                              <1> ; DMA Registers (for 'sysdma')
620                              <1> ; 02/07/2017 (sb16mod.s)
621 000149EA 00020406C0C4C8CC    <1> dma_adr:    db 0,2,4,6,0C0h,0C4h,0C8h,0CCh
622 000149F2 01030507C2C6CACE    <1> dma_cnt:    db 1,3,5,7,0C2h,0C6h,0CAh,0CEh
623 000149FA 878381828F8B898A    <1> dma_page:   db 87h,83h,81h,82h,8Fh,8Bh,89h,8Ah ; 03/08/2017
624 00014A02 0A0A0A0AD4D4D4D4    <1> dma_mask:   db 0Ah,0Ah,0Ah,0Ah,0D4h,0D4h,0D4h,0D4h
625 00014A0A 0B0B0B0BD6D6D6D6    <1> dma_mod:    db 0Bh,0Bh,0Bh,0Bh,0D6h,0D6h,0D6h,0D6h
626 00014A12 0C0C0C0CD8D8D8D8    <1> dma_flip:   db 0Ch,0Ch,0Ch,0Ch,0D8h,0D8h,0D8h,0D8h
3096
3097                                     ; 27/08/2014
3098                                     scr_row:
3099 00014A1A E0810B00                         dd 0B8000h + 0A0h + 0A0h + 0A0h ; Row 3
3100                                     scr_col:
3101 00014A1E 00000000                         dd 0
3102
3103 00014A22 90<rept>                   Align 4
3104                                           ; 15/04/2016
3105                                           ; TRDOS 386 (TRDOS v2.0)
3106
3107                                           ; 21/08/2014
3108                                     ilist:
3109                                           ;times       32 dd cpu_except ; INT 0 to INT 1Fh
3110                                           ;
3111                                           ; Exception list
3112                                           ; 25/08/2014
3113 00014A24 [EA0B0000]                       dd    exc0   ; 0h,  Divide-by-zero Error
3114 00014A28 [F10B0000]                       dd    exc1
3115 00014A2C [F80B0000]                       dd    exc2
3116 00014A30 [FF0B0000]                       dd    exc3
3117 00014A34 [030C0000]                       dd    exc4
3118 00014A38 [070C0000]                       dd    exc5
3119 00014A3C [0B0C0000]                       dd    exc6   ; 06h,  Invalid Opcode
3120 00014A40 [0F0C0000]                       dd    exc7
3121 00014A44 [130C0000]                       dd    exc8
3122 00014A48 [170C0000]                       dd    exc9
3123 00014A4C [1B0C0000]                       dd    exc10
3124 00014A50 [1F0C0000]                       dd    exc11
3125 00014A54 [230C0000]                       dd    exc12
3126 00014A58 [270C0000]                       dd    exc13  ; 0Dh, General Protection Fault
```

```
00014A5C [2B0C0000]                     dd      exc14  ; 0Eh, Page Fault
00014A60 [2F0C0000]                     dd      exc15
00014A64 [330C0000]                     dd      exc16
00014A68 [370C0000]                     dd      exc17
00014A6C [3B0C0000]                     dd      exc18
00014A70 [3F0C0000]                     dd      exc19
00014A74 [430C0000]                     dd      exc20
00014A78 [470C0000]                     dd      exc21
00014A7C [4B0C0000]                     dd      exc22
00014A80 [4F0C0000]                     dd      exc23
00014A84 [530C0000]                     dd      exc24
00014A88 [570C0000]                     dd      exc25
00014A8C [5B0C0000]                     dd      exc26
00014A90 [5F0C0000]                     dd      exc27
00014A94 [630C0000]                     dd      exc28
00014A98 [670C0000]                     dd      exc29
00014A9C [6B0C0000]                     dd      exc30
00014AA0 [6F0C0000]                     dd      exc31
                                IRQ_list: ; 28/02/2017 ('syscalbac')
                                        ; Interrupt list
00014AA4 [5E090000]                     dd      timer_int     ; INT 20h
                                                ;dd    irq0
00014AA8 [D6100000]                     dd      kb_int        ; 24/01/2016
                                                ;dd    irq1
00014AAC [400B0000]                     dd      irq2
                                                ; COM2 int
00014AB0 [440B0000]                     dd      irq3
                                                ; COM1 int
00014AB4 [4F0B0000]                     dd      irq4
00014AB8 [5A0B0000]                     dd      irq5
                                ;DISKETTE_INT: ;06/02/2015
00014ABC [DB510000]                     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)
00014AC0 [C90E0000]                     dd      default_irq7 ; 25/02/2015
                                                ;dd    irq7
                                ; Real Time Clock Interrupt
00014AC4 [C90A0000]                     dd      rtc_int             ; 23/02/2015, IRQ 8 handler
                                                ;dd    irq8   ; INT 28h
00014AC8 [6A0B0000]                     dd      irq9
00014ACC [6E0B0000]                     dd      irq10
00014AD0 [720B0000]                     dd      irq11
00014AD4 [760B0000]                     dd      irq12
00014AD8 [7A0B0000]                     dd      irq13
                                ;HDISK_INT1:  ;06/02/2015
00014ADC [8E5B0000]                     dd      hdc1_int      ; 21/02/2015, IRQ 14 handler
                                                ;dd    irq14
                                ;HDISK_INT2:  ;06/02/2015
00014AE0 [B55B0000]                     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

00014AE4 [414B0100]                     dd      int30h        ; Reserved for
                                                              ; !!! Retro UNIX (RUNIX) !!!
                                                              ; !!! SINGLIX !!! System Calls
00014AE8 [C9170000]                     dd      int31h        ; Video BIOS (IBM PC/AT, Int 10h)
00014AEC [F50E0000]                     dd      int32h        ; Keyboard Functions (IBM PC/AT, Int 16h)
00014AF0 [92520000]                     dd      int33h        ; DISK I/O (IBM PC/AT, Int 13h)
00014AF4 [3B2D0100]                     dd      int34h        ; #IOCTL# (I/O port access support for ring 3)
00014AF8 [466A0000]                     dd      int35h        ; Time/Date Functions (IBM PC/AT, Int 1Ah)
00014AFC [7D0D0000]                     dd      ignore_int    ; INT 36h : Timer Functions
00014B00 [7D0D0000]                     dd      ignore_int    ; INT 37h
00014B04 [7D0D0000]                     dd      ignore_int    ; INT 38h
00014B08 [7D0D0000]                     dd      ignore_int    ; INT 39h
00014B0C [7D0D0000]                     dd      ignore_int    ; INT 3Ah
00014B10 [7D0D0000]                     dd      ignore_int    ; INT 3Bh
00014B14 [7D0D0000]                     dd      ignore_int    ; INT 3Ch
00014B18 [7D0D0000]                     dd      ignore_int    ; INT 3Dh
00014B1C [7D0D0000]                     dd      ignore_int    ; INT 3Eh
00014B20 [7D0D0000]                     dd      ignore_int    ; INT 3Fh
00014B24 [BAD70000]                     dd      sysent        ; INT 40h : !!! TRDOS 386 System Calls !!!
                                                ;dd    ignore_int
00014B28 00000000                       dd      0

                                ; 20/08/2014
                                  ; /* This is the default interrupt "handler" :-) */
                                  ; Linux v0.12 (head.s)
                                int_msg:
00014B2C 556E6B6E6F776E2069-            db "Unknown interrupt ! ", 0
00014B35 6E7465727275707420-
00014B3E 212000

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

                                ; 29/04/2016
                                int30h:
                                trdos_isc_routine:
                                        ; 02/05/2016
                                        ; 01/05/2016
                                        ; 29/04/2016
                                        ; 18/04/2016
                                        ; 15/04/2016 (TRDOS 386 = TRDOS v2.0)
                                        ; 17/04/2011 (TRDOS v1.0, 'IFC.ASM')
                                        ; 03/02/2011 ('trdos_ifc_routine')
                                        ;
00014B41 8B1C24                         mov     ebx, [esp] ; EIP (next)
00014B44 83EB02                         sub     ebx, 2 ; EIP (CD ##h)

00014B47 89C1                           mov     ecx, eax
```

```
3230 00014B49 8A4301                    mov   al, [ebx+1] ; CDh ##h
3231
3232 00014B4C 66BA1000                  mov   dx, KDATA
3233 00014B50 8EDA                      mov   ds, dx
3234 00014B52 8EC2                      mov   es, dx
3235
3236 00014B54 FC                        cld
3237 00014B55 8B15[188A0100]            mov   edx, [k_page_dir]
3238 00014B5B 0F22DA                    mov   cr3, edx
3239
3240 00014B5E E872F7FEFF                call  bytetohex
3241 00014B63 66A3[97470100]            mov   [int_num_str], ax
3242
3243 00014B69 89D8                      mov   eax, ebx ; EIP
3244 00014B6B E8A5F7FEFF                call  dwordtohex
3245 00014B70 8915[B3470100]            mov   [eip_str], edx
3246 00014B76 A3[B7470100]              mov   [eip_str+4], eax
3247
3248 00014B7B 89C8                      mov   eax, ecx
3249 00014B7D E893F7FEFF                call  dwordtohex
3250 00014B82 8915[A2470100]            mov   [eax_str], edx
3251 00014B88 A3[A6470100]              mov   [eax_str+4], eax
3252
3253 00014B8D 43                        inc   ebx
3254 00014B8E 8A03                      mov   al, [ebx] ; Interrupt number
3255
3256                                trdos_isc_handler:
3257 00014B90 80FE30                    cmp   dh, 30h ; Retro UNIX, SINGLIX System calls
3258 00014B93 7507                      jne   short trdos_usi_handler
3259 00014B95 BE[34470100]              mov   esi, isc_msg
3260 00014B9A EB05                      jmp   short loc_write_inv_system_call_msg
3261
3262                                trdos_usi_handler:
3263 00014B9C BE[4A470100]              mov   esi, usi_msg
3264
3265                                loc_write_inv_system_call_msg:
3266 00014BA1 E8BF29FFFF                call  print_msg
3267                                    ; 29/04/2016
3268 00014BA6 BE[80470100]              mov   esi, inv_msg_for_trdos_v2
3269 00014BAB E8B529FFFF                call  print_msg
3270
3271                                loc_ifc_terminate_process:
3272                                    ; u.uno = process number
3273                                    ; 29/04/2016
3274
3275                                    ; 02/05/2016
3276 00014BB0 FE05[5B030300]            inc   byte [sysflg] ; 0FFh -> 0
3277
3278 00014BB6 B801000000                mov   eax, 1
3279 00014BBB E9D38EFFFF                jmp   sysexit
3280
3281                                ; 07/03/2015
3282                                ; Temporary Code
3283                                display_disks:
3284 00014BC0 803D[0E6E0000]00          cmp   byte [fd0_type], 0
3285 00014BC7 7605                      jna   short ddsks1
3286 00014BC9 E87D000000                call  pdskm
3287                                ddsks1:
3288 00014BCE 803D[0F6E0000]00          cmp   byte [fd1_type], 0
3289 00014BD5 760C                      jna   short ddsks2
3290 00014BD7 C605[1B4D0100]31          mov   byte [dskx], '1'
3291 00014BDE E868000000                call  pdskm
3292                                ddsks2:
3293 00014BE3 803D[106E0000]00          cmp   byte [hd0_type], 0
3294 00014BEA 7654                      jna   short ddsk6
3295 00014BEC 66C705[194D0100]68-       mov   word [dsktype], 'hd'
3295 00014BF4 64
3296 00014BF5 C605[1B4D0100]30          mov   byte [dskx], '0'
3297 00014BFC E84A000000                call  pdskm
3298                                ddsks3:
3299 00014C01 803D[116E0000]00          cmp   byte [hd1_type], 0
3300 00014C08 7636                      jna   short ddsk6
3301 00014C0A C605[1B4D0100]31          mov   byte [dskx], '1'
3302 00014C11 E835000000                call  pdskm
3303                                ddsks4:
3304 00014C16 803D[126E0000]00          cmp   byte [hd2_type], 0
3305 00014C1D 7621                      jna   short ddsk6
3306 00014C1F C605[1B4D0100]32          mov   byte [dskx], '2'
3307 00014C26 E820000000                call  pdskm
3308                                ddsks5:
3309 00014C2B 803D[136E0000]00          cmp   byte [hd3_type], 0
3310 00014C32 760C                      jna   short ddsk6
3311 00014C34 C605[1B4D0100]33          mov   byte [dskx], '3'
3312 00014C3B E80B000000                call  pdskm
3313                                ddsk6:
3314 00014C40 BE[434D0100]              mov   esi, nextline
3315 00014C45 E806000000                call  pdskml
3316                                pdskm_ok:
3317 00014C4A C3                        retn
3318                                pdskm:
3319 00014C4B BE[174D0100]              mov   esi, dsk_ready_msg
3320                                pdskml:
3321 00014C50 AC                        lodsb
3322 00014C51 08C0                      or    al, al
3323 00014C53 74F5                      jz    short pdskm_ok
3324 00014C55 56                        push  esi
3325                                    ; 13/05/2016
3326 00014C56 BB07000000                  mov     ebx, 7  ; Black background,
3327                                                    ; light gray forecolor
3328                                                    ; Video page 0 (bh=0)
3329 00014C5B E8A5D6FEFF                call  _write_tty
3330 00014C60 5E                        pop   esi
3331 00014C61 EBED                      jmp   short pdskml
3332
3333 00014C63 90                    Align 2
```

```
                                        ; 21/08/2014
                                exc_msg:
00014C64 435055206578636570-            db "CPU exception ! "
00014C6D 74696F6E202120
                                excnstr:            ; 25/08/2014
00014C74 3F3F68202045495020-            db "??h", "  EIP : "
00014C7D 3A20
                                EIPstr: ; 29/08/2014
00014C7F 00<rept>                       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:
00014C8B 07                             db      07h
00014C8C 0D0A                           db      0Dh, 0Ah
                                        ;db     "MEMORY ALLOCATION INFO", 0Dh, 0Ah, 0Dh, 0Ah
00014C8E 546F74616C206D656D-            db      "Total memory : "
00014C97 6F7279203A20
                                mem_total_b_str: ; 10 digits
00014C9D 303030303030303030-            db      "0000000000 bytes", 0Dh, 0Ah
00014CA6 302062797465730D0A
00014CAF 202020202020202020-            db      "                ", 20h, 20h, 20h
00014CB8 202020202020202020
                                mem_total_p_str: ; 7 digits
00014CC1 303030303030302070-            db      "0000000 pages", 0Dh, 0Ah
00014CCA 616765730D0A
00014CD0 0D0A                           db      0Dh, 0Ah
00014CD2 46726565206D656D6F-            db      "Free memory  : "
00014CDB 727920203A20
                                free_mem_b_str:  ; 10 digits
00014CE1 3F3F3F3F3F3F3F3F3F-            db      "?????????? bytes", 0Dh, 0Ah
00014CEA 3F2062797465730D0A
00014CF3 202020202020202020-            db      "                ", 20h, 20h, 20h
00014CFC 202020202020202020
                                free_mem_p_str:  ; 7 digits
00014D05 3F3F3F3F3F3F3F2070-            db      "??????? pages", 0Dh, 0Ah
00014D0E 616765730D0A
00014D14 0D0A00                         db      0Dh, 0Ah, 0

                                dsk_ready_msg:
00014D17 0D0A                           db      0Dh, 0Ah
                                dsktype:
00014D19 6664                           db      'fd'
                                dskx:
00014D1B 30                             db      '0'
00014D1C 20                             db      20h
00014D1D 697320524541445920-            db      'is READY ...'
00014D26 2E2E2E
00014D29 00                             db      0

                                setup_error_msg:
00014D2A 0D0A                           db 0Dh, 0Ah
00014D2C 4469736B2053657475-            db 'Disk Setup Error !'
00014D35 70204572726F722021
00014D3E 0D0A00                         db 0Dh, 0Ah,0

                                next2line: ; 08/02/2016
00014D41 0D0A                           db      0Dh, 0Ah
                                nextline:
00014D43 0D0A00                         db      0Dh, 0Ah, 0

                                ; temporary
                                ; 19/12/2020
                                msg_lfb_addr:
                                        ;db     0Dh, 0Ah
00014D46 4C696E656172206672-            db      "Linear frame buffer at "
00014D4F 616D65206275666665-
00014D58 7220617420
                                lfb_addr_str: ; 8 (hex) digits
00014D5D 303030303030303068-            db      "00000000h", 0Dh, 0Ah
00014D66 0D0A
00014D68 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:
00014D6B 0D0A07                         db 0Dh, 0Ah, 07h
00014D6E 4552524F523A204B65-            db 'ERROR: Kernel Panic !'
00014D77 726E656C2050616E69-
00014D80 632021
00014D83 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:
00014D86 5475726B6973682052-               db "Turkish Rational DOS v2.0 [02/03/2021] ...", 0
00014D8F 6174696F6E616C2044-
00014D98 4F532076322E30205B-
00014DA1 30322F30332F323032-
00014DAA 315D202E2E2E00
                                    NextLine:
00014DB1 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   03FFFFFFh ; 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
00014DB4 B886801524             <1>       mov     eax, (ICH_DID << 16) + INTEL_VID
00014DB9 E876000000             <1>         call    pciFindDevice
00014DBE 730D                   <1>         jnc     short d_ac97_1
                                <1> d_ac97_0:
                                <1> ; couldn't find the audio device!
00014DC0 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
00014DC1 B806115930             <1>       mov     eax, (VT8233_DID << 16) + VIA_VID
00014DC6 E869000000             <1>         call    pciFindDevice
                                <1> ;       jnc     short d_vt8233_0
                                <1> ; couldn't find the audio device!
                                <1> ;     retn
00014DCB 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>
00014DCD A3[A09C0100]           <1>       mov   [audio_dev_id], eax
00014DD2 8915[A49C0100]         <1>       mov   [audio_vendor], edx
                                <1>
                                <1>       ; init controller
00014DD8 B004                   <1>       mov   al, PCI_CMD_REG ; command register (04h)
00014DDA E8E2000000             <1>       call  pciRegRead32
                                <1>
```

```
                                <1>       ; eax = BUS/DEV/FN/REG
                                <1>       ; edx = STATUS/COMMAND
                                <1>       ;     SSSSSSSSSSSSSSSSCCCCCCCCCCCCCCCC
00014DDF 8915[A89C0100]         <1>       mov   [audio_stats_cmd], edx
                                <1>
00014DE5 B010                   <1>       mov   al, PCI_IO_BASE ; IO base address register (10h)
                                <1>       ;mov  al, NAMBAR_REG      ; Native Audio Mixer BAR (10h)
00014DE7 E8D5000000             <1>       call  pciRegRead32
                                <1>
00014DEC 66813D[A49C0100]86-    <1>       cmp   word [audio_vendor], 8086h ; AC'97 ?
00014DF4 80                     <1>
00014DF5 751F                   <1>       jne   short d_vt8233_1
                                <1>
00014DF7 6683E2FE               <1>       and   dx, 0FFFEh ; Audio Codec IO_ADDR_MASK
00014DFB 668915[D09C0100]       <1>       mov   [NAMBAR], dx
                                <1>
00014E02 B014                   <1>       mov   al, NABMBAR_REG ; Native Audio Bus Mastering BAR (14h)
00014E04 E8B8000000             <1>       call  pciRegRead32
                                <1>
00014E09 6683E2C0               <1>       and   dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00014E0D 668915[D29C0100]       <1>       mov   [NABMBAR], dx
                                <1>         ;mov [audio_io_base], dx
                                <1>
00014E14 EB0B                   <1>       jmp   short d_ac97_2
                                <1>
                                <1> d_vt8233_1:
00014E16 6683E2C0               <1>       and    dx, 0FFC0h ; Audio Controller IO_ADDR_MASK
00014E1A 668915[9E9C0100]       <1>         mov    [audio_io_base], dx
                                <1>
                                <1> d_ac97_2:
                                <1>       ; 10/06/2017
00014E21 B03C                   <1>       mov   al, AC97_INT_LINE ; Interrupt Line Register (3Ch)
                                <1>       ;call pciRegRead32
00014E23 E886000000             <1>       call  pciRegRead8
                                <1>
                                <1>       ;and  edx, 0FFh
00014E28 6681E2FF00             <1>       and   dx, 0FFh
                                <1>
00014E2D 8815[9B9C0100]         <1>       mov    [audio_intr], dl
                                <1>
00014E33 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
00014E34 50             <1>       push   eax
                        <1>       ;push  esi
                        <1>       ;push  edi
                        <1>
00014E35 89C6           <1>         mov     esi, eax                ; save off vend+device ID
00014E37 BF00FFFF7F     <1>         mov     edi, (80000000h - 100h) ; start with bus 0, dev 0 func 0
                        <1>
                        <1> nextPCIdevice:
00014E3C 81C700010000   <1>         add     edi, 100h
00014E42 81FF00F8FF80   <1>         cmp     edi, 80FFF800h         ; scanned all devices?
00014E48 F9             <1>         stc
00014E49 740C           <1>         je      short PCIScanExit       ; not found
                        <1>
00014E4B 89F8           <1>         mov     eax, edi                ; read PCI registers
00014E4D E86F000000     <1>         call    pciRegRead32
00014E52 39F2           <1>         cmp     edx, esi                ; found device?
00014E54 75E6           <1>         jne     short nextPCIdevice
00014E56 F8             <1>         clc
                        <1>
                        <1> PCIScanExit:
00014E57 9C             <1>       pushf
00014E58 B8FFFFFF7F     <1>       mov    eax, NOT_BIT31      ; 19/03/2017
00014E5D 21F8           <1>       and    eax, edi     ; return only bus/dev/fn #
00014E5F 9D             <1>       popf
                        <1>
                        <1>       ;pop   edi
                        <1>       ;pop   esi
00014E60 5A             <1>       pop    edx
                        <1>       ;pop   ecx
00014E61 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>
00014E62 53             <1>       push   ebx
00014E63 51             <1>       push   ecx
00014E64 89C3           <1>         mov     ebx, eax        ; save eax, dh
00014E66 88F1           <1>         mov     cl, dh
                        <1>
00014E68 25FFFFFF3F     <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
00014E6D 0D00000080     <1>         or      eax, BIT31             ; make a PCI access request
00014E72 24FC           <1>         and     al, ~3 ; NOT 3         ; force index to be dword
                        <1>
00014E74 66BAF80C       <1>         mov     dx, PCI_INDEX_PORT
00014E78 EF             <1>         out  dx, eax                   ; write PCI selector
                        <1>
00014E79 66BAFC0C       <1>         mov     dx, PCI_DATA_PORT
00014E7D 88D8           <1>         mov     al, bl
00014E7F 2403           <1>         and     al, 3                  ; figure out which port to
00014E81 00C2           <1>         add     dl, al                 ; read to
                        <1>
00014E83 F7C3000000C0   <1>       test    ebx, PCI32+PCI16
00014E89 7507           <1>         jnz     short _pregr0
00014E8B EC             <1>       in     al, dx             ; return 8 bits of data
00014E8C 88C2           <1>         mov  dl, al
00014E8E 88CE           <1>       mov     dh, cl                  ; restore dh for 8 bit read
00014E90 EB12           <1>       jmp    short _pregr2
                        <1> _pregr0:
00014E92 F7C300000080   <1>       test    ebx, PCI32
00014E98 7507           <1>         jnz  short _pregr1
00014E9A 66ED           <1>       in     ax, dx
00014E9C 6689C2         <1>         mov     dx, ax                 ; return 16 bits of data
00014E9F EB03           <1>       jmp    short _pregr2
                        <1> _pregr1:
00014EA1 ED             <1>       in     eax, dx                  ; return 32 bits of data
00014EA2 89C2           <1>       mov    edx, eax
                        <1> _pregr2:
00014EA4 89D8           <1>       mov     eax, ebx          ; restore eax
00014EA6 25FFFFFF3F     <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
00014EAB 59             <1>       pop    ecx
00014EAC 5B             <1>       pop    ebx
00014EAD C3             <1>       retn
                        <1>
                        <1> pciRegRead8:
00014EAE 25FFFFFF3F     <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit read size
```

```
513 00014EB3 EBAD                <1>         jmp     short pciRegRead; call generic PCI access
514                              <1>
515                              <1> pciRegRead16:
516 00014EB5 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit read size
517 00014EBA 0D00000040          <1>         or      eax, PCI16             ; call generic PCI access
518 00014EBF EBA1                <1>         jmp     short pciRegRead
519                              <1>
520                              <1> pciRegRead32:
521 00014EC1 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit read size
522 00014EC6 0D00000080          <1>         or      eax, PCI32             ; call generic PCI access
523 00014ECB EB95                <1>         jmp     pciRegRead
524                              <1>
525                              <1> pciRegWrite:
526                              <1>       ; 03/04/2017 ('pci.asm', 29/11/2016)
527                              <1>       ;
528                              <1>       ; 8/16/32bit PCI writer
529                              <1>       ;
530                              <1>       ; Entry: EAX=PCI Bus/Device/fn/register number
531                              <1>       ;           BIT31 set if 32 bit access requested
532                              <1>       ;           BIT30 set if 16 bit access requested
533                              <1>       ;           otherwise defaults to 8bit read
534                              <1>       ;        DL/DX/EDX data to write depending on size
535                              <1>       ;
536                              <1>       ; Note1: this routine is meant to be called via pciRegWrite8,
537                              <1>       ;       pciRegWrite16 or pciRegWrite32 as detailed below.
538                              <1>       ;
539                              <1>       ; Note2: don't attempt to write 32bits of data from a non dword
540                              <1>       ;       aligned reg number. Likewise, don't do 16 bit writes from
541                              <1>       ;       non word aligned reg #
542                              <1>
543 00014ECD 53                  <1>       push   ebx
544 00014ECE 51                  <1>       push   ecx
545 00014ECF 89C3                <1>         mov     ebx, eax        ; save eax, edx
546 00014ED1 89D1                <1>         mov     ecx, edx
547 00014ED3 25FFFFFF3F          <1>       and     eax, NOT_PCI32_PCI16     ; clear out data size request
548 00014ED8 0D00000080          <1>         or      eax, BIT31             ; make a PCI access request
549 00014EDD 24FC                <1>         and     al, ~3 ; NOT 3         ; force index to be dword
550                              <1>
551 00014EDF 66BAF80C            <1>         mov     dx, PCI_INDEX_PORT
552 00014EE3 EF                  <1>         out  dx, eax                   ; write PCI selector
553                              <1>
554 00014EE4 66BAFC0C            <1>         mov     dx, PCI_DATA_PORT
555 00014EE8 88D8                <1>         mov     al, bl
556 00014EEA 2403                <1>         and     al, 3                  ; figure out which port to
557 00014EEC 00C2                <1>         add     dl, al                 ; write to
558                              <1>
559 00014EEE F7C3000000C0        <1>       test    ebx, PCI32+PCI16
560 00014EF4 7505                <1>         jnz     short _pregw0
561 00014EF6 88C8                <1>       mov    al, cl                    ; put data into al
562 00014EF8 EE                  <1>       out    dx, al
563 00014EF9 EB12                <1>       jmp    short _pregw2
564                              <1> _pregw0:
565 00014EFB F7C300000080        <1>       test    ebx, PCI32
566 00014F01 7507                <1>         jnz     short _pregw1
567 00014F03 6689C8              <1>       mov    ax, cx             ; put data into ax
568 00014F06 66EF                <1>       out    dx, ax
569 00014F08 EB03                <1>       jmp    short _pregw2
570                              <1> _pregw1:
571 00014F0A 89C8                <1>       mov    eax, ecx           ; put data into eax
572 00014F0C EF                  <1>       out    dx, eax
573                              <1> _pregw2:
574 00014F0D 89D8                <1>         mov     eax, ebx        ; restore eax
575 00014F0F 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; clear out data size request
576 00014F14 89CA                <1>         mov     edx, ecx        ; restore dx
577 00014F16 59                  <1>       pop    ecx
578 00014F17 5B                  <1>       pop    ebx
579 00014F18 C3                  <1>       retn
580                              <1>
581                              <1> pciRegWrite8:
582 00014F19 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 8 bit write size
583 00014F1E EBAD                <1>         jmp  short pciRegWrite   ; call generic PCI access
584                              <1>
585                              <1> pciRegWrite16:
586 00014F20 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 16 bit write size
587 00014F25 0D00000040          <1>         or      eax, PCI16             ; call generic PCI access
588 00014F2A EBA1                <1>         jmp  short pciRegWrite
589                              <1>
590                              <1> pciRegWrite32:
591 00014F2C 25FFFFFF3F          <1>         and     eax, NOT_PCI32_PCI16   ; set up 32 bit write size
592 00014F31 0D00000080          <1>         or      eax, PCI32             ; call generic PCI access
593 00014F36 EB95                <1>         jmp  pciRegWrite
594                              <1>
595                              <1> init_codec:
596                              <1>       ; 05/06/2017
597                              <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
598                              <1>       ;
599 00014F38 A1[A09C0100]        <1>       mov    eax, [audio_dev_id]
600 00014F3D B041                <1>       mov    al, VIA_ACLINK_CTRL
601 00014F3F E86AFFFFFF          <1>       call   pciRegRead8
602                              <1>       ; ?
603 00014F44 B040                <1>       mov    al, VIA_ACLINK_STAT
604 00014F46 E863FFFFFF          <1>       call   pciRegRead8
605 00014F4B F6C201              <1>       test   dl, VIA_ACLINK_C00_READY
606 00014F4E 7508                <1>         jnz     short _codec_ready_1
607 00014F50 E80E000000          <1>       call   reset_codec
608 00014F55 7306                <1>       jnc    short _codec_ready_2 ; eax = 1
609 00014F57 C3                  <1>       retn
610                              <1> _codec_ready_1:
611 00014F58 B801000000          <1>       mov    eax, 1
612                              <1> _codec_ready_2:
613 00014F5D E886000000          <1>       call   codec_io_w16
614                              <1> detect_codec:
615 00014F62 C3                  <1>       retn
616                              <1>
617                              <1> reset_codec:
```

```
                        <1>       ; 16/04/2017
                        <1>       ; 23/03/2017
                        <1>       ; ('codec.asm')
                        <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
00014F63 A1[A09C0100]   <1>       mov    eax, [audio_dev_id]
00014F68 B041           <1>       mov    al, VIA_ACLINK_CTRL
00014F6A B2E0           <1>              mov    dl, VIA_ACLINK_CTRL_ENABLE + VIA_ACLINK_CTRL_RESET + VIA_ACLINK_CTRL_SYNC
00014F6C E8A8FFFFFF     <1>       call   pciRegWrite8
                        <1>
00014F71 E849000000     <1>       call   delay_100ms  ; wait 100 ms
                        <1> _rc_cold:
00014F76 E814000000     <1>         call    cold_reset
00014F7B 7301           <1>         jnc     short _reset_codec_ok
                        <1>
                        <1>       ; 16/04/2017
                        <1>         ;xor     eax, eax         ; timeout error
                        <1>             ;stc
00014F7D C3             <1>       retn
                        <1>
                        <1> _reset_codec_ok:
                        <1>       ; 01/09/2020
                        <1>       ; 15/08/2020
                        <1>       ; 27/07/2020
                        <1>       ; also reset codec by using index control register 0 of AD1980 or ALC655
                        <1>       ; (to fix line out -2 channels audio playing- problem on AD1980 codec)
                        <1>
00014F7E 29C0           <1>       sub    eax, eax
00014F80 BA00000000     <1>       mov    edx, CODEC_RESET_REG ; 00h ; Reset register
00014F85 E8CA000000     <1>       call   codec_write
                        <1>
                        <1>       ;sub   eax, eax
                        <1>       ; 01/09/2020
                        <1>       ; 15/08/2020
                        <1>       ; AD1980 BugFix
                        <1>       ; (set HPSEL -headphone amp to be driven from mixer- and
                        <1>       ;      CLDIS - center and LFE disable- bits)
                        <1>       ;mov   eax, 0C00h ; HPSEL = bit 10, CLDIS = bit 11 ; 01/09/2020
                        <1>       ;mov   edx, CODEC_MISC_CRTL_BITS_REG ; 76h ; Misc Ctrl Bits ; AD1980
                        <1>       ;call  codec_write
                        <1>
00014F8A 31C0           <1>         xor     eax, eax
                        <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
00014F8C FEC0           <1>         inc  al
00014F8E C3             <1>       retn
                        <1>
                        <1> cold_reset:
                        <1>       ; 16/04/2017
                        <1>       ; 23/03/2017
                        <1>       ; ('codec.asm')
                        <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                        <1>       ;mov   eax, [audio_dev_id]
                        <1>       ;mov   al, VIA_ACLINK_CTRL
00014F8F 30D2           <1>       xor    dl, dl ; 0
00014F91 E883FFFFFF     <1>       call   pciRegWrite8
                        <1>
00014F96 E824000000     <1>       call   delay_100ms  ; wait 100 ms
                        <1>
                        <1>       ;; ACLink on, deassert ACLink reset, VSR, SGD data out
                        <1>         ;; note - FM data out has trouble with non VRA codecs !!
                        <1>
                        <1>       ;mov   eax, [audio_dev_id]
                        <1>       ;mov   al, VIA_ACLINK_CTRL
00014F9B B2CC           <1>       mov    dl, VIA_ACLINK_CTRL_INIT
00014F9D E877FFFFFF     <1>       call   pciRegWrite8
                        <1>
00014FA2 B910000000     <1>       mov    ecx, 16      ; total 2s
                        <1>
                        <1> _crst_wait:
                        <1>       ;mov   eax, [audio_dev_id]
00014FA7 B040           <1>       mov    al, VIA_ACLINK_STAT
00014FA9 E800FFFFFF     <1>       call   pciRegRead8
                        <1>
00014FAE F6C201         <1>         test    dl, VIA_ACLINK_C00_READY
00014FB1 750B           <1>         jnz     short _crst_ok
                        <1>
00014FB3 51             <1>       push   ecx
00014FB4 E806000000     <1>       call   delay_100ms
00014FB9 59             <1>       pop    ecx
                        <1>
00014FBA 49             <1>         dec     ecx
00014FBB 75EA           <1>         jnz     short _crst_wait
                        <1>
                        <1> _crst_fail:
00014FBD F9             <1>         stc
                        <1> _crst_ok:
00014FBE C3             <1>       retn
                        <1>
                        <1> delay_100ms:
                        <1>       ; 29/05/2017
                        <1>       ; 24/03/2017 ('codec.asm')
                        <1>       ; wait 100 ms
00014FBF B990010000     <1>       mov    ecx, 400  ; 400*0.25ms
                        <1> _delay_x_ms:
00014FC4 E803000000     <1>       call   delay1_4ms
00014FC9 E2F9           <1>         loop _delay_x_ms
00014FCB C3             <1>       retn
                        <1>
                        <1> ;       delay1_4ms - Delay for 1/4 millisecond.
                        <1> ;         1mS = 1000us
                        <1> ;       Entry:
                        <1> ;         None
                        <1> ;       Exit:
                        <1> ;       None
                        <1> ;
                        <1> ;       Modified:
```

```
                        <1> ;         None
                        <1> ;
                        <1>
                        <1>       ; 29/05/2017
                        <1>       ; 23/04/2017
                        <1>       ; 05/03/2017 (TRDOS 386)
                        <1>       ; ('UTILS.ASM')
                        <1> delay1_4ms:
00014FCC 50             <1>         push    eax
00014FCD 51             <1>         push    ecx
00014FCE B110           <1>         mov  cl, 16       ; close enough.
                        <1>
00014FD0 E461           <1>       in     al, PORTB ; 61h
                        <1>
00014FD2 2410           <1>       and    al, REFRESH_STATUS ; 10h
00014FD4 88C5           <1>       mov    ch, al       ; Start toggle state
                        <1> _d4ms1:
00014FD6 E461           <1>       in     al, PORTB    ; Read system control port
                        <1>
00014FD8 2410           <1>       and    al, REFRESH_STATUS ; Refresh toggles 15.085 microseconds
00014FDA 38C5           <1>       cmp    ch, al
00014FDC 74F8           <1>       je     short _d4ms1 ; Wait for state change
                        <1>
00014FDE 88C5           <1>       mov    ch, al       ; Update with new state
00014FE0 FEC9           <1>       dec    cl
00014FE2 75F2           <1>       jnz    short _d4ms1
                        <1>
00014FE4 F8             <1>       clc    ; 29/05/2017
                        <1>
00014FE5 59             <1>         pop     ecx
00014FE6 58             <1>         pop     eax
00014FE7 C3             <1>         retn
                        <1>
                        <1> ; 10/04/2017 (TRDOS 386)
                        <1> ; 12/11/2016
                        <1>
                        <1> codec_io_w16: ;w32
                        <1>       ; ('codec.asm')
00014FE8 668B15[9E9C0100]   <1>         mov  dx, [audio_io_base]
00014FEF 6681C28000         <1>         add     dx, VIA_REG_AC97
00014FF4 EF                 <1>       out    dx, eax
00014FF5 C3                 <1>         retn
                        <1>
                        <1> codec_io_r16: ;r32
                        <1>       ; ('codec.asm')
00014FF6 668B15[9E9C0100]   <1>         mov     dx, [audio_io_base]
00014FFD 6681C28000         <1>         add     dx, VIA_REG_AC97
00015002 ED                 <1>         in   eax, dx
00015003 C3                 <1>         retn
                        <1>
                        <1> ctrl_io_w8:
                        <1>       ; ('codec.asm')
00015004 660315[9E9C0100]   <1>         add     dx, [audio_io_base]
0001500B EE                 <1>         out  dx, al
0001500C C3                 <1>         retn
                        <1>
                        <1> ctrl_io_r8:
                        <1>       ; ('codec.asm')
0001500D 660315[9E9C0100]   <1>         add     dx, [audio_io_base]
00015014 EC                 <1>         in   al, dx
00015015 C3                 <1>         retn
                        <1>
                        <1> ctrl_io_w32:
                        <1>       ; ('codec.asm')
00015016 660315[9E9C0100]   <1>         add     dx, [audio_io_base]
0001501D EF                 <1>         out  dx, eax
0001501E C3                 <1>         retn
                        <1>
                        <1> ctrl_io_r32:
                        <1>       ; ('codec.asm')
0001501F 660315[9E9C0100]   <1>         add  dx, [audio_io_base]
00015026 ED                 <1>       in     eax, dx
00015027 C3                 <1>         retn
                        <1>
                        <1> codec_read:
                        <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                        <1>         ; Use only primary codec.
                        <1>         ; eax = register
00015028 C1E010             <1>         shl     eax, VIA_REG_AC97_CMD_SHIFT
0001502B 0D00008002         <1>         or      eax, VIA_REG_AC97_PRIMARY_VALID + VIA_REG_AC97_READ
                        <1>
00015030 E8B3FFFFFF         <1>       call    codec_io_w16
                        <1>
                        <1>             ; codec_valid
00015035 E831000000         <1>       call   codec_check_ready
0001503A 7301               <1>         jnc  short _cr_ok
                        <1>
0001503C C3                 <1>       retn
                        <1>
                        <1> _cr_ok:
                        <1>       ; wait 25 ms
0001503D B950000000         <1>       mov    ecx, 80 ; (100*0.25 ms)
                        <1> _cr_wloop:
00015042 E885FFFFFF         <1>       call   delay1_4ms
00015047 E2F9               <1>       loop   _cr_wloop
                        <1>
00015049 E8A8FFFFFF         <1>         call    codec_io_r16
0001504E 25FFFF0000         <1>         and     eax, 0FFFFh
00015053 C3                 <1>         retn
                        <1>
                        <1> codec_write:
                        <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                        <1>         ; Use only primary codec.
                        <1>
                        <1>       ; eax = data (volume)
```

```
                        <1>       ; edx = register (mixer register)
                        <1>
00015054 C1E210         <1>       shl     edx, VIA_REG_AC97_CMD_SHIFT
                        <1>
00015057 C1E000         <1>         shl     eax, VIA_REG_AC97_DATA_SHIFT ; shl eax, 0
0001505A 09C2           <1>         or      edx, eax
                        <1>
0001505C B800000000     <1>         mov     eax, VIA_REG_AC97_CODEC_ID_PRIMARY
00015061 C1E01E         <1>         shl     eax, VIA_REG_AC97_CODEC_ID_SHIFT
00015064 09D0           <1>         or      eax, edx
                        <1>
00015066 E87DFFFFFF     <1>         call    codec_io_w16
                        <1>         ;mov    [codec.regs+esi], ax
                        <1>
                        <1>         ;call      codec_check_ready
                        <1>             ;retn
                        <1>       ;jmp   short _codec_check_ready
                        <1>
                        <1> codec_check_ready:
                        <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
                        <1>
                        <1> _codec_check_ready:
0001506B B914000000     <1>       mov    ecx, 20      ; total 2s
                        <1> _ccr_wait:
00015070 51             <1>       push   ecx
                        <1>
00015071 E880FFFFFF     <1>         call    codec_io_r16
00015076 A900000001     <1>         test    eax, VIA_REG_AC97_BUSY
0001507B 740B           <1>         jz      short _ccr_ok
                        <1>
0001507D E83DFFFFFF     <1>       call   delay_100ms
                        <1>
00015082 59             <1>       pop    ecx
                        <1>
00015083 49             <1>       dec     ecx
00015084 75EA           <1>         jnz     short _ccr_wait
                        <1>
00015086 F9             <1>         stc
00015087 C3             <1>         retn
                        <1>
                        <1> _ccr_ok:
00015088 59             <1>       pop    ecx
00015089 25FFFF0000     <1>       and     eax, 0FFFFh
0001508E C3             <1>         retn
                        <1>
                        <1> codec_config:
                        <1>       ; 10/06/2017
                        <1>       ; 29/05/2017
                        <1>       ; 24/04/2017
                        <1>       ; 21/04/2017
                        <1>       ; 16/04/2017 (TRDOS 386 Kernel)
                        <1>       ; 15/11/2016 ('codec.asm', 'player.com')
                        <1>       ; 14/11/2016
                        <1>       ; 12/11/2016 - Erdogan Tan
                        <1>       ;           (Ref: KolibriOS, 'setup_codec', codec.inc)
                        <1>
0001508F B802020000     <1>       mov     eax, 0202h
00015094 66A3[CE9C0100] <1>       mov    [audio_master_volume], ax
0001509A 66B81F1F       <1>       mov    ax, 1F1Fh ; 31,31
0001509E BA02000000     <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
000150A3 E8ACFFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0202h
000150A8 66B80202       <1>       mov     ax, 0202h
000150AC BA18000000     <1>       mov    edx, CODEC_PCM_OUT_REG ; 18h ; Wave Output (Stereo)
000150B1 E89EFFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0202h
000150B6 66B80202       <1>       mov    ax, 0202h
000150BA BA04000000     <1>       mov    edx, CODEC_AUX_VOL ; 04h ; CODEC_HP_VOL_REG ; HeadPhone
000150BF E890FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 08h
                        <1>         ;mov    ax,  08h
000150C4 66B80880       <1>       mov    ax, 8008h ; Mute
000150C8 BA0C000000     <1>       mov    edx, 0Ch  ; AC97_PHONE_VOL ; TAD Input (Mono)
000150CD E882FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0808h
000150D2 66B80808       <1>         mov     ax,  0808h
000150D6 BA10000000     <1>         mov  edx, CODEC_LINE_IN_VOL_REG ; 10h ; Line Input (Stereo)
000150DB E874FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0808h
000150E0 66B80808       <1>       mov     ax,  0808h
000150E4 BA12000000     <1>         mov  edx, CODEC_CD_VOL_REG ; 12h ; CR Input (Stereo)
000150E9 E866FFFFFF     <1>       call   codec_write
                        <1>       ;jc    short cconfig_error
                        <1>
                        <1>       ;mov    eax, 0808h
000150EE 66B80808       <1>       mov     ax,  0808h
000150F2 BA16000000     <1>         mov  edx, CODEC_AUX_VOL_REG ; 16h ; Aux Input (Stereo)
                        <1>       ;call  codec_write
                        <1>       ;;jc   short cconfig_error
000150F7 E958FFFFFF     <1>       jmp    codec_write ; 10/06/2017
                        <1>
                        <1> ;     ; Extended Audio Status (2Ah)
                        <1> ;     mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
                        <1> ;     call   codec_read
                        <1> ;     and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
```

```
                                <1> ;       ;or     eax, 1                  ; set VRA (BIT0)
                                <1> ;       or      eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
                                <1> ;       mov     edx, CODEC_EXT_AUDIO_CTRL_REG
                                <1> ;       call    codec_write
                                <1> ;       ;jc     short cconfig_error
                                <1> ;
                                <1> ;set_sample_rate:
                                <1> ;       ;movzx  eax, word [audio_freq]
                                <1> ;       mov     ax, [audio_freq]
                                <1> ;       mov     edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
                                <1> ;       ;call   codec_write
                                <1> ;       ;retn
                                <1> ;       jmp     codec_write
                                <1>
                                <1> ;cconfig_error:
                                <1> ;       retn
                                <1>
                                <1> vt8233_int_handler:
                                <1>         ; 27/07/2020
                                <1>         ; 22/07/2020
                                <1>         ; Interrupt Handler for VIA VT8237R Audio Controller
                                <1>         ; Note: called by 'dev_IRQ_service'
                                <1>         ; 14/10/2017
                                <1>         ; 09/10/2017, 10/10/2017, 12/10/2017
                                <1>         ; 13/06/2017
                                <1>         ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>         ; 24/03/2017 - 'PLAYER.COM' ('player.asm')
                                <1>
                                <1>         ;push  eax ; * must be saved !
                                <1>         ;push  edx
                                <1>         ;push  ecx
                                <1>         ;push  ebx ; * must be saved !
                                <1>         ;push  esi
                                <1>         ;push  edi
                                <1>
                                <1>         ;cmp   byte [audio_busy], 1
                                <1>         ;jnb   short _ih0 ; 09/10/2017
                                <1>
                                <1>         ;mov   byte [audio_flag_eol], 0
                                <1>
000150FC 66BA0000               <1>           mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
00015100 E808FFFFFF             <1>           call    ctrl_io_r8
                                <1>
00015105 A880                   <1>         test    al, VIA_REG_STAT_ACTIVE
00015107 7417                   <1>           jz      short _ih0 ; 09/10/2017
                                <1>
00015109 2407                   <1>           and     al, VIA_REG_STAT_EOL + VIA_REG_STAT_FLAG + VIA_REG_STAT_STOPPED
0001510B A2[CD9C0100]           <1>         mov   [audio_flag_eol], al
00015110 740E                   <1>           jz   short _ih0 ; 09/10/2017
                                <1>
                                <1>         ; 09/10/2017
                                <1>         ;mov   byte [audio_busy], 1
                                <1>
00015112 803D[CC9C0100]01       <1>         cmp    byte [audio_play_cmd], 1
00015119 7315                   <1>         jnb    short _ih1 ; 10/10/2017
                                <1>
0001511B E84A000000             <1>         call   channel_reset
                                <1> _ih0:
                                <1>         ; 09/10/2017
00015120 A0[CD9C0100]           <1>           mov     al, [audio_flag_eol]   ;; ack ;;
00015125 66BA0000               <1>           mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
00015129 E8D6FEFFFF             <1>           call    ctrl_io_w8
0001512E EB39                   <1>         jmp    short _ih4
                                <1> _ih1:
                                <1> vt8233_tuneLoop:
00015130 A0[CD9C0100]           <1>           mov     al, [audio_flag_eol]   ;; ack ;;
00015135 66BA0000               <1>           mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STATUS
00015139 E8C6FEFFFF             <1>           call    ctrl_io_w8
                                <1>
                                <1>         ; 22/07/2020
                                <1>         ;; 12/10/2017
                                <1>         ;mov   byte [audio_flag], 0 ; Reset
                                <1>
                                <1>         ; 10/10/2017
                                <1>         ; 09/10/2017
                                <1>         ;test  byte [audio_flag_eol], VIA_REG_STAT_FLAG
                                <1>         ;jz    short _ih2 ; EOL
                                <1>
                                <1>         ; 22/07/2020
                                <1>         ; 14/10/2017
                                <1>         ;test  byte [audio_flag_eol], VIA_REG_STAT_EOL
                                <1>         ;jnz   short _ih2 ; EOL
                                <1>         ;               ; (Half Buffer 2 has been completed
                                <1>         ;               ; and Half Buffer 1 will be played.)
                                <1>
                                <1>         ; FLAG
                                <1>         ; (Half Buffer 1 has been completed
                                <1>         ;  and Half Buffer 2 will be played.)
                                <1>
                                <1>         ; 14/10/2017
                                <1>         ;; (Continue to play.)
                                <1>         ;mov   al, VIA_REG_CTRL_INT
                                <1>                ;or    al, VIA_REG_CTRL_START
                                <1>                ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
                                <1>           ;call      ctrl_io_w8
                                <1>         ; 12/10/2017
                                <1>         ;mov   byte [audio_flag], 1
                                <1>
                                <1>         ; 22/07/2020
                                <1>         ;inc   byte [audio_flag] ; = 1
                                <1> _ih2:
                                <1>         ; 10/10/2017
0001513E 8B3D[B89C0100]         <1>         mov    edi, [audio_dma_buff]
00015144 8B0D[BC9C0100]         <1>         mov    ecx, [audio_dmabuff_size]
0001514A D1E9                   <1>         shr    ecx, 1 ; dma buff size / 2 = half buffer size
```

```
1038                                    <1>
1039                                    <1>       ; 22/07/2020
1040                                    <1>       ; 12/10/2017
1041                                    <1>       ;cmp   byte [audio_flag], 0
1042                                    <1>       ;ja    short _ih3 ; Playing Half Buffer 2 (Current: FLAG)
1043                                    <1>
1044                                    <1>       ; 27/07/2020
1045                                    <1>       ; 22/07/2020
1046 0001514C F605[C09C0100]01         <1>       test   byte [audio_flag], 1  ; Current flag value
1047 00015153 7402                     <1>       jz     short _ih3 ; Half Buffer 1 must be filled
1048                                    <1>
1049                                    <1>       ; Half Buffer 2 must be filled
1050 00015155 01CF                     <1>       add    edi, ecx
1051                                    <1> _ih3:
1052                                    <1>       ; Update half buffer 2 while playing half buffer 1
1053                                    <1>       ; Update half buffer 1 while playing half buffer 2
1054                                    <1>
1055 00015157 8B35[B09C0100]           <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
1056 0001515D C1E902                   <1>       shr    ecx, 2 ; half buff size / 4
1057 00015160 F3A5                     <1>       rep    movsd
1058                                    <1>
1059                                    <1>       ; switch flag value ;
1060 00015162 8035[C09C0100]01         <1>       xor    byte [audio_flag], 1
1061                                    <1>       ; 12/10/2017
1062                                    <1>       ; [audio_flag] = 0 : Playing dma half buffer 2
1063                                    <1>                          ; Next buffer (to update) is dma half buff 1
1064                                    <1>       ;             = 1 : Playing dma half buffer 1
1065                                    <1>                          ; Next buffer (to update) is dma half buff 2
1066                                    <1> _ih4:
1067                                    <1>       ; 28/05/2017
1068                                    <1>       ;mov   byte [audio_busy], 0 ; 09/10/2017
1069                                    <1>       ;
1070                                    <1>       ;pop   edi
1071                                    <1>       ;pop   esi
1072                                    <1>       ;pop   ebx ; * must be restored !
1073                                    <1>       ;pop   ecx
1074                                    <1>       ;pop   edx
1075                                    <1>       ;pop   eax ; * must be restored !
1076                                    <1>
1077 00015169 C3                       <1>       retn
1078                                    <1>
1079                                    <1> channel_reset:
1080                                    <1>       ; 24/06/2017
1081                                    <1>       ; 29/05/2017
1082                                    <1>       ; 23/03/2017
1083                                    <1>       ; 14/11/2016 - Erdogan Tan
1084                                    <1>       ; 12/11/2016 - Erdogan Tan (Ref: KolibriOS, vt823x.asm)
1085 0001516A BA01000000               <1>         mov  edx, VIA_REG_OFFSET_CONTROL
1086                                    <1>         ;mov eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE + VIA_REG_CTRL_RESET
1087 0001516F B848000000               <1>         mov  eax, VIA_REG_CTRL_PAUSE + VIA_REG_CTRL_TERMINATE ; 24/06/2017
1088 00015174 E88BFEFFFF               <1>       call    ctrl_io_w8
1089                                    <1>
1090                                    <1>         ;mov edx, VIA_REG_OFFSET_CONTROL
1091                                    <1>         ;call   ctrl_io_r8
1092                                    <1>
1093                                    <1>       ; wait for 50 ms
1094 00015179 B9A0000000               <1>       mov    ecx, 160 ; (200*0.25 ms) ; 29/05/2017
1095                                    <1> _ch_rst_wait:
1096 0001517E E849FEFFFF               <1>       call   delay1_4ms
1097 00015183 49                       <1>       dec    ecx
1098 00015184 75F8                     <1>       jnz    short _ch_rst_wait
1099                                    <1>
1100                                    <1>         ; disable interrupts
1101 00015186 BA01000000               <1>         mov  edx, VIA_REG_OFFSET_CONTROL
1102 0001518B 31C0                     <1>         xor     eax, eax
1103 0001518D E872FEFFFF               <1>         call    ctrl_io_w8
1104                                    <1>
1105                                    <1>         ; clear interrupts
1106 00015192 BA00000000               <1>         mov  edx, VIA_REG_OFFSET_STATUS
1107 00015197 B803000000               <1>       mov    eax, 3
1108 0001519C E863FEFFFF               <1>         call ctrl_io_w8
1109                                    <1>
1110                                    <1>       ;mov   edx, VIA_REG_OFFSET_CURR_PTR
1111                                    <1>       ;xor   eax, eax
1112                                    <1>       ;call  ctrl_io_w32
1113                                    <1>
1114 000151A1 C3                       <1>         retn
1115                                    <1>
1116                                    <1> vt8233_stop: ; 22/04/2017
1117 000151A2 C605[CC9C0100]00         <1>       mov    byte [audio_play_cmd], 0 ; stop !
1118                                    <1> _tlp2:
1119                                    <1>       ; 24/06/2017
1120                                    <1>         ; finished with song, stop everything
1121                                    <1>       ;mov   al, VIA_REG_CTRL_INT
1122                                    <1>         ;or  al, VIA_REG_CTRL_TERMINATE
1123                                    <1>       ;mov   dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1124                                    <1>         ;call      ctrl_io_w8
1125                                    <1>
1126                                    <1>         ;call      channel_reset
1127                                    <1>       ;retn
1128                                    <1>
1129 000151A9 EBBF                     <1>       jmp    short channel_reset
1130                                    <1>
1131                                    <1> set_vt8233_bdl: ; Set VT8237R Buffer Descriptor List
1132                                    <1>       ; 22/07/2020 - TRDOS 386 v2.0.2
1133                                    <1>       ; 28/05/2017
1134                                    <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
1135                                    <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
1136                                    <1>
1137                                    <1>       ; eax = dma buffer address = [audio_DMA_buff]
1138                                    <1>       ; ecx = dma buffer buffer size = [audio_dmabuff_size]
1139                                    <1>
1140 000151AB D1E9                     <1>       shr    ecx, 1 ; dma half buffer size
1141 000151AD 89CE                     <1>       mov    esi, ecx
1142                                    <1>
```

```
1143 000151AF BF[D49C0100]        <1>         mov     edi, audio_bdl_buff    ; get BDL address
1144 000151B4 B910000000          <1>         mov     ecx, 32 / 2            ; make 32 entries in BDL
1145                              <1>
1146 000151B9 EB05                <1>       jmp    short s_vt8233_bdl1
1147                              <1>
1148                              <1> s_vt8233_bdl0:
1149                              <1>       ; set buffer descriptor 0 to start of data file in memory
1150                              <1>
1151 000151BB A1[B89C0100]        <1>       mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
1152                              <1>
1153                              <1> s_vt8233_bdl1:
1154 000151C0 AB                  <1>       stosd                    ; store dmabuffer1 address
1155                              <1>
1156 000151C1 89C2                <1>       mov    edx, eax
1157                              <1>
1158                              <1> ; VIA VT8235.PDF: (Page 110) (Erdogan Tan, 29/11/2016)
1159                              <1>       ;
1160                              <1>       ;      Audio SGD Table Format
1161                              <1>       ;      --------------------------------
1162                              <1>       ;      63   62    61-56    55-32  31-0
1163                              <1>       ;      --   --   --------  -----  ----
1164                              <1>       ;      EOL FLAG -reserved- Base   Base
1165                              <1>       ;                        Count  Address
1166                              <1>       ;                         [23:0] [31:0]
1167                              <1>       ;      EOL: End Of Link.
1168                              <1>       ;           1 indicates this block is the last of the link.
1169                              <1>       ;           If the channel "Interrupt on EOL" bit is set, then
1170                              <1>       ;           an interrupt is generated at the end of the transfer.
1171                              <1>       ;
1172                              <1>       ;      FLAG: Block Flag. If set, transfer pauses at the end of this
1173                              <1>       ;            block. If the channel "Interrupt on FLAG" bit is set,
1174                              <1>       ;            then an interrupt is generated at the end of this block.
1175                              <1>
1176 000151C3 89F0                <1>       mov    eax, esi ; DMA half buffer size
1177 000151C5 01C2                <1>       add    edx, eax
1178 000151C7 0D00000040          <1>       or     eax, FLAG
1179                              <1>       ;or    eax, EOL
1180 000151CC AB                  <1>       stosd
1181                              <1>
1182                              <1> ; 2nd buffer:
1183                              <1>
1184 000151CD 89D0                <1>         mov  eax, edx ; Physical address of the 2nd half of DMA buffer
1185 000151CF AB                  <1>       stosd         ; store dmabuffer2 address
1186                              <1>
1187                              <1> ; set length to [audio_dmabuff_size]/2
1188                              <1> ; Set control (bits 31:16) to BUP, bits 15:0=number of samples
1189                              <1> ;
1190 000151D0 89F0                <1>       mov    eax, esi ; DMA half buffer size
1191                              <1>       ; 22/07/2020
1192                              <1>       ;or    eax, EOL
1193 000151D2 0D00000040          <1>       or     eax, FLAG
1194 000151D7 AB                  <1>       stosd
1195                              <1>
1196 000151D8 E2E1                <1>       loop    s_vt8233_bdl0
1197                              <1>
1198                              <1>       ; 22/07/2020
1199 000151DA 814FFC00000080      <1>       or     dword [edi-4], EOL
1200                              <1>
1201 000151E1 C3                  <1>       retn
1202                              <1>
1203                              <1> vt8233_start_play:
1204                              <1>       ; 01/09/2020
1205                              <1>       ; 22/07/2020
1206                              <1>       ; start to play audio data via VT8233 audio controller
1207                              <1>       ; 13/06/2017
1208                              <1>       ; 10/06/2017
1209                              <1>       ; 24/04/2017
1210                              <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
1211                              <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
1212                              <1>       ; write buffer descriptor list address
1213                              <1>
1214                              <1>       ; Extended Audio Status (2Ah)
1215 000151E2 B82A000000          <1>       mov    eax, CODEC_EXT_AUDIO_CTRL_REG ; 2Ah
1216 000151E7 E83CFEFFFF          <1>       call   codec_read
1217 000151EC 25FDFF0000          <1>       and     eax, 0FFFFh - 2          ; clear DRA (BIT1)
1218                              <1>       ;or     eax, 1                   ; set VRA (BIT0)
1219                              <1>       ;or    eax, 5       ; VRA (BIT0) & S/PDIF (BIT2) ; 14/11/2016
1220 000151F1 0C05                <1>       or     al, 5
1221                              <1>       ; 01/09/2020
1222                              <1>       ;or    eax, 3805h ; AD1980 (PRK, PRJ, PRI = 1 .. only front DAC)
1223                              <1>       ; 01/09/2020
1224                              <1>       ;mov   edx, CODEC_EXT_AUDIO_CTRL_REG
1225                              <1>       ;cmp   word [audio_freq], 0BB80h ; 48 kHz
1226                              <1>       ;jne   short set_extd_audio_status_1
1227                              <1>       ;and   al, 0FEh ; disable VRA bit (set sample rate to 48000 Hz)
1228                              <1>       ;jmp   short set_extd_audio_status_2
1229                              <1> ;set_extd_audio_status_1:
1230 000151F3 BA2A000000          <1>       mov    edx, CODEC_EXT_AUDIO_CTRL_REG
1231 000151F8 E857FEFFFF          <1>       call   codec_write
1232                              <1>       ;jc    short cconfig_error
1233                              <1>
1234                              <1> set_sample_rate:
1235                              <1>       ;movzx eax, word [audio_freq]
1236 000151FD 66A1[CA9C0100]      <1>       mov    ax, [audio_freq]
1237 00015203 BA2C000000          <1>       mov    edx, CODEC_PCM_FRONT_DACRATE_REG ; 2Ch ; PCM Front DAC Rate
1238                              <1> ;set_extd_audio_status_2:
1239 00015208 E847FEFFFF          <1>       call   codec_write
1240                              <1>
1241                              <1>       ; 01/09/2020
1242                              <1>       ; set AD1980 MCB register (Index 76h) to 0C00h
1243                              <1>       ; (CLDIS, HPSEL)
1244                              <1>       ;mov   ax, 0C00h
1245                              <1>       ;mov   edx, CODEC_MISC_CRTL_BITS_REG ; 76h
1246                              <1>       ;                  ; Miscellaneous Control Bit Register
1247                              <1>       ;call  codec_write
```

```
1248                                    <1>       ;
1249                                    <1>
1250 0001520D B8[D49C0100]              <1>         mov  eax, audio_bdl_buff
1251                                    <1>
1252                                    <1>       ; 12/11/2016 - Erdogan Tan
1253                                    <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
1254 00015212 BA04000000               <1>       mov    edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_TABLE_PTR
1255 00015217 E8FAFDFFFF               <1>         call ctrl_io_w32
1256                                    <1>
1257                                    <1>       ;call  codec_check_ready
1258                                    <1>
1259 0001521C 66BA0200                 <1>       mov    dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_L
1260                                    <1>         ;mov eax, 2 ; 31
1261 00015220 B01F                     <1>       mov    al, 31
1262 00015222 2A05[CE9C0100]           <1>         sub  al, [audio_master_volume_l]
1263 00015228 E8D7FDFFFF               <1>       call   ctrl_io_w8
1264                                    <1>
1265                                    <1>       ;call  codec_check_ready
1266                                    <1>
1267 0001522D 66BA0300                 <1>         mov     dx, VIADEV_PLAYBACK + VIA_REG_OFS_PLAYBACK_VOLUME_R
1268                                    <1>         ;mov ax, 2  ; 31
1269 00015231 B01F                     <1>       mov    al, 31
1270 00015233 2A05[CF9C0100]           <1>         sub  al, [audio_master_volume_r]
1271 00015239 E8C6FDFFFF               <1>       call    ctrl_io_w8
1272                                    <1>
1273                                    <1>       ;call  codec_check_ready
1274                                    <1> ;
1275                                    <1> ;
1276                                    <1> ; All set. Let's play some music.
1277                                    <1> ;
1278                                    <1> ;
1279                                    <1>             ;mov    dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
1280                                    <1>         ;mov    ax, VIA8233_REG_TYPE_16BIT or VIA8233_REG_TYPE_STEREO or 0xfffff or 0xff000000
1281                                    <1>         ;call   ctrl_io_w32
1282                                    <1>
1283                                    <1>       ;call  codec_check_ready
1284                                    <1>
1285                                    <1>       ; 08/12/2016
1286                                    <1>       ; 07/10/2016
1287                                    <1>         ;;mov    al, 1
1288                                    <1>         ;mov al, 31
1289                                    <1>       ; 22/07/2020
1290 0001523E B0FF                     <1>       mov    al, 0FFh
1291 00015240 E813000000               <1>       call    set_VT8233_LastValidIndex
1292                                    <1>
1293 00015245 C605[CC9C0100]01         <1>       mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
1294                                    <1>
1295                                    <1>       ; 22/07/2020
1296                                    <1>       ;mov  byte [audio_flag], 0  ; clear half buffer flag
1297                                    <1>
1298                                    <1> vt8233_play: ; continue to play
1299                                    <1>       ; 22/04/2017
1300                                    <1>       ;mov  al, VIA_REG_CTRL_INT
1301                                    <1>             ;or    al, VIA_REG_CTRL_START
1302                                    <1>         ;;mov      al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START
1303                                    <1>       ; 22/07/2020
1304 0001524C B0A1                     <1>       mov    al, VIA_REG_CTRL_AUTOSTART + VIA_REG_CTRL_START + VIA_REG_CTRL_INT_FLAG
1305                                    <1>
1306 0001524E 66BA0100                 <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1307 00015252 E8ADFDFFFF               <1>         call    ctrl_io_w8
1308                                    <1>       ;call  codec_check_ready
1309                                    <1>       ;retn
1310                                    <1>       ;jmp   codec_check_ready
1311 00015257 C3                       <1>       retn
1312                                    <1>
1313                                    <1> ;input AL = index # to stop on
1314                                    <1> set_VT8233_LastValidIndex:
1315                                    <1>       ; 23/07/2020
1316                                    <1>       ; 10/06/2017
1317                                    <1>       ; 21/04/2017 (TRDOS 386 kernel, 'audio.s')
1318                                    <1>       ; 24/03/2017 - 'PLAYER.COM' ('via_wav.asm' - 29/11/2016)
1319                                    <1>       ; 19/11/2016
1320                                    <1>       ; 14/11/2016 - Erdogan Tan (Ref: VIA VT8235.PDF, Page 110)
1321                                    <1>       ; 12/11/2016 - Erdogan Tan
1322                                    <1>       ; (Ref: KolibriOS, vt823x.asm, 'create_primary_buff')
1323                                    <1>       ;push  edx
1324                                    <1>       ;push  ax
1325 00015258 50                       <1>       push   eax ; 23/07/2020
1326                                    <1>       ;push  ecx
1327 00015259 0FB705[CA9C0100]         <1>       movzx  eax, word [audio_freq] ; Hertz
1328 00015260 BA00001000               <1>       mov    edx, 100000h ; 2^20 = 1048576
1329 00015265 F7E2                     <1>       mul    edx
1330 00015267 B980BB0000               <1>       mov    ecx, 48000
1331 0001526C F7F1                     <1>       div    ecx
1332                                    <1>       ;and   eax, 0FFFFFh
1333                                    <1>       ;pop   ecx
1334                                    <1>       ;pop   dx
1335 0001526E 5A                       <1>       pop    edx ; 23/07/2020
1336 0001526F C1E218                   <1>       shl    edx, 24  ; STOP Index Setting: Bit 24 to 31
1337 00015272 09D0                     <1>       or     eax, edx
1338                                    <1>       ; 19/11/2016
1339 00015274 803D[C89C0100]10         <1>       cmp    byte [audio_bps], 16
1340 0001527B 7505                     <1>       jne    short sLVI_1
1341 0001527D 0D00002000               <1>       or     eax, VIA8233_REG_TYPE_16BIT
1342                                    <1> sLVI_1:
1343 00015282 803D[C99C0100]02         <1>       cmp    byte [audio_stmo], 2
1344 00015289 7505                     <1>       jne    short sLVI_2
1345 0001528B 0D00001000               <1>       or     eax, VIA8233_REG_TYPE_STEREO
1346                                    <1> sLVI_2:
1347 00015290 BA08000000               <1>       mov     edx, VIADEV_PLAYBACK + VIA_REG_OFFSET_STOP_IDX
1348 00015295 E87CFDFFFF               <1>         call    ctrl_io_w32
1349                                    <1>       ;call  codec_check_ready
1350                                    <1>       ;pop   edx
1351 0001529A C3                       <1>       retn
1352                                    <1>
```

```
1353                                    <1> vt8233_pause: ; pause
1354                                    <1>       ; 10/06/2017
1355                                    <1>       ; 22/04/2017
1356                                    <1>       ;mov     al, VIA_REG_CTRL_INT
1357                                    <1>         ;or      al, VIA_REG_CTRL_PAUSE
1358                                    <1>       ; 23/07/2020
1359 0001529B B029                     <1>       mov    al, VIA_REG_CTRL_PAUSE+VIA_REG_CTRL_INT_FLAG+VIA_REG_CTRL_AUTOSTART
1360                                    <1>
1361 0001529D 66BA0100                 <1>       mov     dx, VIADEV_PLAYBACK + VIA_REG_OFFSET_CONTROL
1362 000152A1 E85EFDFFFF               <1>         call    ctrl_io_w8
1363                                    <1>       ;call  codec_check_ready
1364                                    <1>       ;retn
1365                                    <1>       ;jmp   codec_check_ready
1366 000152A6 C3                       <1>       retn
1367                                    <1>
1368                                    <1> vt8233_reset:
1369                                    <1>       ; 22/04/2017
1370                                    <1>       ; reset VT8237R (vt8233) Audio Controller
1371                                    <1>       ;cmp   byte [audio_play_cmd], 1
1372                                    <1>       ;jna   short vt8233_rst_0
1373 000152A7 C605[CC9C0100]00          <1>       mov    byte [audio_play_cmd], 0 ; stop !
1374                                    <1> vt8233_rst_0:
1375 000152AE E8B0FCFFFF               <1>       call   reset_codec
1376 000152B3 720A                     <1>       jc     short vt8233_rst_1 ; codec error !
1377                                    <1>       ; eax = 1
1378 000152B5 E82EFDFFFF               <1>       call   codec_io_w16 ; w32
1379 000152BA E8ABFEFFFF               <1>       call   channel_reset
1380                                    <1> vt8233_rst_1:
1381 000152BF C3                       <1>       retn
1382                                    <1>
1383                                    <1> vt8233_volume:
1384                                    <1>       ; set VT8237R (vt8233) sound volume level
1385                                    <1>       ; 24/04/2017
1386                                    <1>       ; 22/04/2017
1387                                    <1>       ; bl = component (0 = master/playback/lineout volume)
1388                                    <1>       ; cl = left channel volume level (0 to 31)
1389                                    <1>       ; ch = right channel volume level (0 to 31)
1390                                    <1>
1391 000152C0 08DB                     <1>       or     bl, bl
1392 000152C2 7520                     <1>       jnz    short vt8233_vol_1 ; temporary !
1393 000152C4 66B81F1F                 <1>       mov    ax, 1F1Fh ; 31,31
1394 000152C8 38C1                     <1>       cmp    cl, al
1395 000152CA 7718                     <1>       ja     short vt8233_vol_1 ; temporary !
1396 000152CC 38E5                     <1>       cmp    ch, ah
1397 000152CE 7714                     <1>       ja     short vt8233_vol_1 ; temporary !
1398 000152D0 66890D[CE9C0100]         <1>       mov    [audio_master_volume], cx
1399 000152D7 6629C8                   <1>       sub    ax, cx
1400 000152DA BA02000000               <1>       mov    edx, CODEC_MASTER_VOL_REG ; 02h ; Line Out
1401 000152DF E870FDFFFF               <1>       call   codec_write
1402                                    <1> vt8233_vol_1:
1403 000152E4 C3                       <1>       retn
1404                                    <1>
1405                                    <1> ; CODE for SOUND BLASTER 16
1406                                    <1>
1407                                    <1> DetectSB:
1408                                    <1>       ; 24/04/2017
1409                                    <1>       ;pushad
1410                                    <1> ScanPort:
1411 000152E5 66BB1002                 <1>       mov     bx, 210h    ; start scanning ports
1412                                    <1>                         ; 210h, 220h, .. 260h
1413                                    <1> ResetDSP:
1414 000152E9 6689DA                   <1>       mov     dx, bx            ; try to reset the DSP.
1415 000152EC 6683C206                 <1>       add     dx, 06h
1416 000152F0 B001                     <1>       mov    al, 1
1417 000152F2 EE                       <1>       out    dx, al
1418                                    <1>
1419 000152F3 EC                       <1>       in     al, dx
1420 000152F4 EC                       <1>       in     al, dx
1421 000152F5 EC                       <1>       in     al, dx
1422 000152F6 EC                       <1>       in     al, dx
1423                                    <1>
1424 000152F7 30C0                     <1>       xor     al, al
1425 000152F9 EE                       <1>       out    dx, al
1426                                    <1>
1427 000152FA 6683C208                 <1>       add     dx, 08h
1428 000152FE 66B96400                 <1>       mov    cx, 100
1429                                    <1> WaitID:
1430 00015302 EC                       <1>       in     al, dx
1431 00015303 08C0                     <1>       or      al, al
1432 00015305 7804                     <1>       js      short GetID
1433 00015307 E2F9                     <1>       loop    WaitID
1434 00015309 EB0F                     <1>       jmp     short NextPort
1435                                    <1> GetID:
1436 0001530B 6683EA04                 <1>       sub     dx, 04h
1437 0001530F EC                       <1>       in     al, dx
1438 00015310 3CAA                     <1>       cmp     al, 0AAh
1439 00015312 7413                     <1>       je      short Found
1440 00015314 6683C204                 <1>       add     dx, 04h
1441 00015318 E2E8                     <1>       loop    WaitID
1442                                    <1> NextPort:
1443 0001531A 6683C310                 <1>       add     bx, 10h           ; if not response,
1444 0001531E 6681FB6002               <1>       cmp     bx, 260h    ; try the next port.
1445 00015323 76C4                     <1>       jbe     short ResetDSP
1446 00015325 F9                       <1>       stc
1447 00015326 C3                       <1>       retn
1448                                    <1> Found:
1449 00015327 66891D[9E9C0100]         <1>       mov     [audio_io_base], bx      ; SB Port Address Found!
1450                                    <1> ScanIRQ:
1451                                    <1> SetIrqs:
1452 0001532E 28C0                     <1>       sub    al, al ; 0
1453 00015330 A2[969C0100]             <1>       mov    [IRQnum], al ; reset
1454 00015335 A2[9B9C0100]             <1>       mov    [audio_intr], al ; reset
1455                                    <1>
1456                                    <1>       ; ah > 0 -> set IRQ vector
1457                                    <1>       ; al = IRQ number
```

```
1458                                    <1>       ;mov   ax, 103h ; IRQ 3
1459                                    <1>       ;call  set_hardware_int_vector
1460                                    <1>       ;mov   ax, 104h ; IRQ 4
1461                                    <1>       ;call  set_hardware_int_vector
1462 0001533A 66B80501                 <1>       mov    ax, 105h ; IRQ 5
1463 0001533E E8C9DDFFFF               <1>       call   set_hardware_int_vector
1464 00015343 66B80701                 <1>       mov    ax, 107h ; IRQ 7
1465 00015347 E8C0DDFFFF               <1>       call   set_hardware_int_vector
1466                                    <1>
1467 0001534C 668B15[9E9C0100]         <1>       mov     dx, [audio_io_base] ; tells to the SB to
1468 00015353 6683C20C                 <1>       add     dx, 0Ch               ; generate a IRQ!
1469                                    <1> WaitSb:
1470 00015357 EC                       <1>       in     al, dx
1471 00015358 08C0                     <1>       or      al, al
1472 0001535A 78FB                     <1>       js      short WaitSb
1473 0001535C B0F2                     <1>       mov     al, 0F2h
1474 0001535E EE                       <1>       out    dx, al
1475                                    <1>
1476 0001535F 31C9                     <1>       xor     ecx, ecx    ; wait until IRQ level
1477                                    <1> WaitIRQ:
1478 00015361 A0[969C0100]             <1>       mov    al, [IRQnum]
1479 00015366 3C00                     <1>       cmp     al, 0 ; is changed or timeout.
1480 00015368 7706                     <1>       ja     short IrqOk
1481 0001536A 6649                     <1>       dec    cx
1482 0001536C 75F3                     <1>       jnz    short WaitIRQ
1483 0001536E EB15                     <1>       jmp    short RestoreIrqs
1484                                    <1> IrqOk:
1485 00015370 A2[9B9C0100]             <1>       mov    [audio_intr], al ; set
1486 00015375 668B15[9E9C0100]         <1>       mov     dx, [audio_io_base]
1487 0001537C 6683C20E                 <1>       add     dx, 0Eh
1488 00015380 EC                       <1>       in     al, dx ; SB acknowledge.
1489 00015381 B020                     <1>       mov    al, 20h
1490 00015383 E620                     <1>       out    20h, al      ; Hardware acknowledge.
1491                                    <1>
1492                                    <1> RestoreIrqs:
1493                                    <1>       ; ah = 0 -> reset IRQ vector
1494                                    <1>       ; al = IRQ number
1495                                    <1>       ;mov   ax, 3 ; IRQ 3
1496                                    <1>       ;call  set_hardware_int_vector
1497                                    <1>       ;mov   ax, 4 ; IRQ 4
1498                                    <1>       ;call  set_hardware_int_vector
1499 00015385 66B80500                 <1>       mov    ax, 5 ; IRQ 5
1500 00015389 E87EDDFFFF               <1>       call   set_hardware_int_vector
1501 0001538E 66B80700                 <1>       mov    ax, 7 ; IRQ 7
1502 00015392 E875DDFFFF               <1>       call   set_hardware_int_vector
1503                                    <1>
1504 00015397 31D2                     <1>       xor    edx, edx
1505 00015399 8915[A09C0100]           <1>       mov    [audio_dev_id], edx ; 0
1506 0001539F 8915[A49C0100]           <1>       mov    [audio_vendor], edx ; 0
1507 000153A5 8915[A89C0100]           <1>       mov    [audio_stats_cmd], edx ; 0
1508                                    <1>
1509                                    <1>       ;popad
1510                                    <1>
1511 000153AB 803D[9B9C0100]01         <1>       cmp     byte [audio_intr], 1 ; IRQ level was changed?
1512                                    <1>
1513 000153B2 C3                       <1>       retn
1514                                    <1>
1515                                    <1> %macro      SbOut  1
1516                                    <1> %%Wait:
1517                                    <1>       in     al, dx
1518                                    <1>       or     al, al
1519                                    <1>       js     short %%Wait
1520                                    <1>       mov    al, %1
1521                                    <1>       out    dx, al
1522                                    <1> %endmacro
1523                                    <1>
1524                                    <1> SbInit_play:
1525                                    <1>       ; 22/10/2017
1526                                    <1>       ; 20/10/2017
1527                                    <1>       ; 06/10/2017
1528                                    <1>       ; 13/07/2017, 09/08/2017
1529                                    <1>       ; 24/04/2017, 15/05/2017, 24/06/2017
1530                                    <1>       ;pushad
1531                                    <1> SetBuffer:
1532                                    <1>       ;mov   byte [DmaFlag], 0
1533                                    <1>
1534 000153B3 8B1D[B89C0100]           <1>       mov    ebx, [audio_dma_buff] ; physical addr of DMA buff
1535 000153B9 89DF                     <1>       mov    edi, ebx
1536 000153BB 8B0D[BC9C0100]           <1>       mov     ecx, [audio_dmabuff_size]
1537                                    <1>
1538 000153C1 803D[C89C0100]10         <1>       cmp    byte [audio_bps], 16
1539 000153C8 7531                     <1>       jne    short sbInit_0 ; set 8 bit DMA buffer
1540                                    <1>
1541                                    <1>       ; 09/08/2017
1542                                    <1>       ; convert byte count to word count
1543 000153CA D1E9                     <1>       shr    ecx, 1
1544 000153CC 49                       <1>       dec    ecx ; word count - 1
1545                                    <1>       ; convert byte offset to word offset
1546 000153CD D1EB                     <1>       shr    ebx, 1
1547                                    <1>
1548                                    <1>       ; 16 bit DMA buffer setting (DMA channel 5)
1549 000153CF B005                     <1>       mov     al, 05h  ; set mask bit for channel 5  (4+1)
1550 000153D1 E6D4                     <1>       out    0D4h, al
1551                                    <1>
1552 000153D3 30C0                     <1>       xor     al, al   ; stops all DMA processes on selected channel
1553 000153D5 E6D8                     <1>       out    0D8h, al ; clear selected channel register
1554                                    <1>
1555 000153D7 88D8                     <1>       mov     al, bl       ; byte 0 of DMA buffer offset in words (physical)
1556 000153D9 E6C4                     <1>       out    0C4h, al ; DMA channel 5 port number
1557                                    <1>
1558 000153DB 88F8                     <1>       mov     al, bh   ; byte 1 of DMA buffer offset in words (physical)
1559 000153DD E6C4                     <1>       out    0C4h, al
1560                                    <1>
1561                                    <1>       ; 09/08/2017
1562 000153DF C1EB0F                   <1>       shr    ebx, 15       ; complete 16 bit shift
```

```
1563 000153E2 80E3FE              <1>       and    bl, 0FEh ; clear bit 0 (not necessary, it will be ignored)
1564                              <1>
1565 000153E5 88D8                <1>       mov     al, bl   ; byte 2 of DMA buffer address (physical)
1566 000153E7 E68B                <1>       out    8Bh, al  ; page register port addr for channel 5 ; 13/07/2017
1567                              <1>
1568 000153E9 88C8                <1>       mov     al, cl   ; low byte of DMA count - 1
1569 000153EB E6C6                <1>       out    0C6h, al ; count register port addr for channel 1
1570                              <1>
1571 000153ED 88E8                <1>       mov     al, ch   ; high byte of DMA count - 1
1572 000153EF E6C6                <1>       out    0C6h, al
1573                              <1>
1574                              <1>       ; channel 5, read, autoinitialized, single mode
1575                              <1>       ;mov   al, 49h
1576 000153F1 B059                <1>       mov    al, 59h  ; 06/10/2017
1577 000153F3 E6D6                <1>       out    0D6h, al ; DMA mode register port address
1578                              <1>
1579 000153F5 B001                <1>       mov     al, 01h  ; clear mask bit for channel 1
1580 000153F7 E6D4                <1>       out    0D4h, al ; DMA mask register port address
1581                              <1>
1582 000153F9 EB28                <1>       jmp    short ClearBuffer
1583                              <1>
1584                              <1> sbInit_0:
1585 000153FB 49                  <1>       dec     ecx ; 09/08/2017
1586                              <1>
1587                              <1>       ; 8 bit DMA buffer setting (DMA channel 1)
1588 000153FC B005                <1>       mov     al, 05h ; set mask bit for channel 1  (4+1)
1589 000153FE E60A                <1>       out    0Ah, al ; DMA mask register
1590                              <1>
1591 00015400 30C0                <1>       xor     al, al  ; stops all DMA processes on selected channel
1592 00015402 E60C                <1>       out    0Ch, al ; clear selected channel register
1593                              <1>
1594 00015404 88D8                <1>       mov     al, bl      ; byte 0 of DMA buffer address (physical)
1595 00015406 E602                <1>       out    02h, al ; DMA channel 1 port number
1596                              <1>
1597 00015408 88F8                <1>       mov     al, bh  ; byte 1 of DMA buffer address (physical)
1598 0001540A E602                <1>       out    02h, al
1599                              <1>
1600 0001540C C1EB10              <1>       shr    ebx, 16
1601                              <1>
1602 0001540F 88D8                <1>       mov     al, bl  ; byte 2 of DMA buffer address (physical)
1603 00015411 E683                <1>       out    83h, al ; page register port addr for channel 1
1604                              <1>
1605 00015413 88C8                <1>       mov     al, cl  ; low byte of DMA count - 1
1606 00015415 E603                <1>       out    03h, al ; count register port addr for channel 1
1607                              <1>
1608 00015417 88E8                <1>       mov     al, ch  ; high byte of DMA count - 1
1609 00015419 E603                <1>       out    03h, al
1610                              <1>
1611                              <1>       ; channel 1, read, autoinitialized, single mode
1612                              <1>       ;mov   al, 49h
1613 0001541B B059                <1>       mov    al, 59h ; 06/10/2017
1614 0001541D E60B                <1>       out    0Bh, al ; DMA mode register port address
1615                              <1>
1616 0001541F B001                <1>       mov     al, 01h ; clear mask bit for channel 1
1617 00015421 E60A                <1>       out    0Ah, al ; DMA mask register port address
1618                              <1>
1619                              <1> ClearBuffer:
1620                              <1>       ;;mov  edi, [audio_dma_buff]
1621                              <1>       ;;mov  ecx, [audio_dmabuff_size]
1622                              <1>       ;inc   ecx
1623                              <1>       ;mov     al, 80h
1624                              <1>       ;;cld
1625                              <1>       ;rep     stosb
1626                              <1> SetIrq:
1627                              <1>       ;mov   ebx, SbIrqhandler
1628                              <1>       ;mov   al, [audio_intr] ; IRQ number
1629                              <1>       ;call  set_dev_IRQ_service
1630                              <1>       ;; SETUP (audio) INTERRUPT CALLBACK SERVICE
1631                              <1>       ;mov   bl, [audio_intr] ; IRQ number
1632                              <1>       ;mov   bh, [audio_cb_mode]
1633                              <1>       ;inc   bh  ; 1 = Signal Response Byte method (fixed value)
1634                              <1>       ;          ; 2 = Callback service method
1635                              <1>       ;          ; 3 = Auto Increment S.R.B. method
1636                              <1>       ;mov   cl, [audio_srb]
1637                              <1>       ;mov   edx, [audio_cb_addr]
1638                              <1>       ;mov   al, [audio_user]
1639                              <1>       ;call  set_irq_callback_service
1640                              <1> ResetDsp:
1641 00015423 668B15[9E9C0100]    <1>       mov     dx, [audio_io_base]
1642 0001542A 6683C206            <1>       add     dx, 06h
1643 0001542E B001                <1>       mov     al, 1
1644 00015430 EE                  <1>       out    dx, al
1645                              <1>
1646 00015431 EC                  <1>       in     al, dx
1647 00015432 EC                  <1>       in     al, dx
1648 00015433 EC                  <1>       in     al, dx
1649 00015434 EC                  <1>       in     al, dx
1650                              <1>
1651 00015435 30C0                <1>       xor     al, al
1652 00015437 EE                  <1>       out    dx, al
1653                              <1>
1654 00015438 66B96400            <1>       mov     cx, 100
1655 0001543C 28E4                <1>       sub    ah, ah ; 0
1656                              <1> WaitId:
1657 0001543E 668B15[9E9C0100]    <1>       mov     dx, [audio_io_base]
1658 00015445 6683C20E            <1>       add     dx, 0Eh
1659 00015449 EC                  <1>       in     al, dx
1660 0001544A 08C0                <1>       or      al, al
1661 0001544C 7807                <1>       js      short sb_GetId
1662 0001544E E2EE                <1>       loop    WaitId
1663 00015450 E9B4000000          <1>       jmp     sb_Exit
1664                              <1> sb_GetId:
1665 00015455 668B15[9E9C0100]    <1>       mov     dx, [audio_io_base]
1666 0001545C 6683C20A            <1>       add     dx, 0Ah
1667 00015460 EC                  <1>       in     al, dx
```

```
1668 00015461 3CAA               <1>       cmp     al, 0AAh
1669 00015463 7407               <1>       je      short SbOk
1670 00015465 E2D7               <1>       loop    WaitId
1671 00015467 E99D000000         <1>       jmp    sb_Exit
1672                             <1> SbOk:
1673 0001546C 668B15[9E9C0100]   <1>       mov     dx, [audio_io_base]
1674 00015473 6683C20C           <1>       add     dx, 0Ch
1675                             <1>       SbOut   0D1h ; Turn on speaker
1675                             <2> %%Wait:
1675 00015477 EC                 <2>  in al, dx
1675 00015478 08C0               <2>  or al, al
1675 0001547A 78FB               <2>  js short %%Wait
1675 0001547C B0D1               <2>  mov al, %1
1675 0001547E EE                 <2>  out dx, al
1676                             <1>       SbOut   41h ; 8 bit or 16 bit transfer
1676                             <2> %%Wait:
1676 0001547F EC                 <2>  in al, dx
1676 00015480 08C0               <2>  or al, al
1676 00015482 78FB               <2>  js short %%Wait
1676 00015484 B041               <2>  mov al, %1
1676 00015486 EE                 <2>  out dx, al
1677 00015487 668B1D[CA9C0100]   <1>       mov    bx, [audio_freq] ; sampling rate (Hz)
1678                             <1>       SbOut  bh ; sampling rate high byte
1678                             <2> %%Wait:
1678 0001548E EC                 <2>  in al, dx
1678 0001548F 08C0               <2>  or al, al
1678 00015491 78FB               <2>  js short %%Wait
1678 00015493 88F8               <2>  mov al, %1
1678 00015495 EE                 <2>  out dx, al
1679                             <1>       SbOut  bl ; sampling rate low byte
1679                             <2> %%Wait:
1679 00015496 EC                 <2>  in al, dx
1679 00015497 08C0               <2>  or al, al
1679 00015499 78FB               <2>  js short %%Wait
1679 0001549B 88D8               <2>  mov al, %1
1679 0001549D EE                 <2>  out dx, al
1680                             <1>
1681                             <1>       ; 22/05/2017
1682 0001549E E8C0000000         <1>       call   sb16_volume_initial ; 15/05/2017
1683                             <1>       ; 20/05/2017
1684                             <1>       ;call  sb16_volume
1685                             <1>
1686                             <1> StartDma:
1687                             <1>       ; autoinitialized mode
1688 000154A3 803D[C89C0100]10   <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
1689 000154AA 7411               <1>       je     short sb_play_1
1690                             <1>       ; 8 bit samples
1691 000154AC 66BBC600           <1>       mov    bx, 0C6h ; 8 bit output (0C6h)
1692 000154B0 803D[C99C0100]02   <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
1693 000154B7 7214               <1>       jb     short sb_play_2
1694 000154B9 B720               <1>       mov    bh, 20h      ; 8 bit stereo (20h)
1695 000154BB EB10               <1>       jmp    short sb_play_2
1696                             <1> sb_play_1:
1697                             <1>       ; 16 bit samples
1698 000154BD 66BBB610           <1>       mov    bx, 10B6h ; 16 bit output (0B6h)
1699 000154C1 803D[C99C0100]02   <1>       cmp    byte [audio_stmo], 2 ; 1 = mono, 2 = stereo
1700 000154C8 7203               <1>       jb     short sb_play_2
1701 000154CA 80C720             <1>       add    bh, 20h      ; 16 bit stereo (30h)
1702                             <1> sb_play_2:
1703                             <1>       ; PCM output (8/16 bit mono autoinitialized transfer)
1704                             <1>       SbOut   bl ; bCommand
1704                             <2> %%Wait:
1704 000154CD EC                 <2>  in al, dx
1704 000154CE 08C0               <2>  or al, al
1704 000154D0 78FB               <2>  js short %%Wait
1704 000154D2 88D8               <2>  mov al, %1
1704 000154D4 EE                 <2>  out dx, al
1705                             <1>       SbOut  bh ; bMode
1705                             <2> %%Wait:
1705 000154D5 EC                 <2>  in al, dx
1705 000154D6 08C0               <2>  or al, al
1705 000154D8 78FB               <2>  js short %%Wait
1705 000154DA 88F8               <2>  mov al, %1
1705 000154DC EE                 <2>  out dx, al
1706 000154DD 8B1D[BC9C0100]     <1>       mov    ebx, [audio_dmabuff_size]  ; 15/05/2017
1707 000154E3 D1EB               <1>       shr    ebx, 1 ; half buffer size
1708                             <1>       ; 20/10/2017
1709 000154E5 803D[C89C0100]10   <1>       cmp    byte [audio_bps], 16 ; 16 bit DMA
1710 000154EC 7502               <1>       jne    short sb_play_3
1711 000154EE D1EB               <1>       shr    ebx, 1 ; byte count to word count
1712                             <1> sb_play_3:
1713 000154F0 664B               <1>       dec    bx  ; wBlkSize is one less than the actual size
1714                             <1>       SbOut   bl
1714                             <2> %%Wait:
1714 000154F2 EC                 <2>  in al, dx
1714 000154F3 08C0               <2>  or al, al
1714 000154F5 78FB               <2>  js short %%Wait
1714 000154F7 88D8               <2>  mov al, %1
1714 000154F9 EE                 <2>  out dx, al
1715                             <1>       SbOut   bh
1715                             <2> %%Wait:
1715 000154FA EC                 <2>  in al, dx
1715 000154FB 08C0               <2>  or al, al
1715 000154FD 78FB               <2>  js short %%Wait
1715 000154FF 88F8               <2>  mov al, %1
1715 00015501 EE                 <2>  out dx, al
1716                             <1>
1717 00015502 C605[CC9C0100]01   <1>       mov    byte [audio_play_cmd], 1 ; playing !
1718                             <1>
1719                             <1>       ;; Set Voice and master volumes
1720                             <1>       ;mov   dx, [audio_io_base]
1721                             <1>       ;add   dl, 4 ; Mixer chip Register Address Port
1722                             <1>       ;SbOut 30h   ; select Master Volume Register (L)
1723                             <1>       ;inc   dl    ; Mixer chip Register Data Port
1724                             <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
```

```
                                <1>       ;dec   dl
                                <1>       ;SbOut 31h   ; select Master Volume Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                                <1>       ;dec   dl
                                <1>       ;SbOut 32h   ; select Voice Volume Register (L)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                                <1>       ;dec   dl
                                <1>       ;SbOut 33h   ; select Voice Volume Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F8h  ; Max. volume value is 31 (31*8)
                                <1>       ;;
                                <1>       ;dec   dl
                                <1>       ;SbOut 44h   ; select Treble Register (L)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
                                <1>       ;dec   dl
                                <1>       ;SbOut 45h   ; select Treble Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Treble value is 15 (15*16)
                                <1>       ;dec   dl
                                <1>       ;SbOut 46h   ; select Bass Register (L)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
                                <1>       ;dec   dl
                                <1>       ;SbOut 47h   ; select Bass Register (R)
                                <1>       ;inc   dl
                                <1>       ;SbOut 0F0h  ; Max. Bass value is 15 (15*16)
                                <1>
                                <1> sb_Exit:
                                <1>       ;popad
00015509 C3                     <1>       retn
                                <1>
                                <1> sb16_int_handler:
                                <1>       ; Interrupt Handler for Sound Blaster 16 Audio Card
                                <1>       ; Note: called by 'dev_IRQ_service'
                                <1>       ; 20/10/2017
                                <1>       ; 12/10/2017
                                <1>       ; 10/10/2017
                                <1>       ; 12/05/2017, 09/10/2017
                                <1>       ; 24/04/2017 (TRDOS 386 kernel, 'audio.s')
                                <1>       ; 10/03/2017 - 'PLAYWAV.PRG' ('playwav.s')
                                <1>
                                <1>       ;push  eax ; * must be saved !
                                <1>       ;push  ebx ; * must be saved !
                                <1>       ;push  ecx
                                <1>       ;push  edx
                                <1>       ;push  esi
                                <1>       ;push  edi
                                <1>
0001550A 668B15[9E9C0100]       <1>       mov     dx, [audio_io_base]
                                <1>       ; 20/10/2017
00015511 80C20F                 <1>       add     dl, 0Fh ; 2xFh (DSP 16 bit intr ack)
00015514 803D[C89C0100]10       <1>       cmp    byte [audio_bps], 16
0001551B 7402                   <1>       je     short sb_irq_16bit_ack
                                <1> sb_irq_8bit_ack:
0001551D FECA                   <1>       dec    dl  ; 2xEh (DSP 8 bit intr ack)
                                <1> sb_irq_16bit_ack:
0001551F EC                     <1>       in     al, dx
                                <1>
                                <1>       ;cmp   byte [audio_busy], 0
                                <1>       ;ja    short sb_irq_h3
                                <1>
                                <1>       ;mov   byte [audio_busy], 1
                                <1>
00015520 803D[CC9C0100]01       <1>       cmp    byte [audio_play_cmd], 1
00015527 7307                   <1>       jnb    short sb_irq_h1
                                <1> sb_irq_h0:
00015529 E8A9000000             <1>       call   sb16_stop
0001552E EB2B                   <1>       jmp    short sb_irq_h3
                                <1> sb_irq_h1:
                                <1>       ;call  sb16_tuneloop
                                <1>       ; 09/10/2017
                                <1> sb16_tuneloop:
00015530 8B3D[B89C0100]         <1>       mov    edi, [audio_dma_buff]
00015536 8B0D[BC9C0100]         <1>       mov    ecx, [audio_dmabuff_size]
0001553C D1E9                   <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>
                                <1>       ; 22/05/2017
0001553E F605[C09C0100]01       <1>       test   byte [audio_flag], 1  ; Current flag value
00015545 7402                   <1>       jz     short sb_tlp1 ; EOL (Half Buffer 1 must be filled)
                                <1>       ; FLAG (Half Buffer 2 must be filled)
00015547 01CF                   <1>       add    edi, ecx
                                <1>       ; 15/05/2017
                                <1> sb_tlp1:
00015549 8B35[B09C0100]         <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
                                <1>       ;rep   movsb
0001554F C1E902                 <1>       shr    ecx, 2 ; half buff size / 4
00015552 F3A5                   <1>       rep    movsd
                                <1>       ;retn
                                <1>
                                <1>       ; 10/10/2017
                                <1>       ; switch flag value
00015554 8035[C09C0100]01       <1>       xor    byte [audio_flag], 1
                                <1>
                                <1>       ; 12/10/2017
                                <1>       ; [audio_flag] = 0 : Playing dma half buffer 2 (odd intr count)
                                <1>                          ; Next buffer (to update) is dma half buff 1
                                <1>       ;              = 1 : Playing dma half buffer 1 (even intr count)
                                <1>                            ; Next buffer (to update) is dma half buff 2
                                <1>
                                <1> sb_irq_h3:
                                <1>       ;mov   byte [audio_busy], 0
                                <1>
```

```
                                <1>       ;pop   edi
                                <1>       ;pop   esi
                                <1>       ;pop   edx
                                <1>       ;pop   ecx
                                <1>       ;pop   ebx ; * must be restored !
                                <1>       ;pop   eax ; * must be restored !
                                <1>
0001555B C3                     <1>       retn
                                <1>
                                <1> sb16_volume:
                                <1>       ; 22/10/2017
                                <1>       ; mov [audio_master_volume_l], cl
                                <1>       ; mov [audio_master_volume_h], ch
0001555C 66890D[CE9C0100]       <1>       mov    [audio_master_volume], cx
                                <1> sb16_volume_initial:
00015563 6652                   <1>       push   dx ; DX (port address) must be saved
00015565 668B15[9E9C0100]       <1>       mov    dx, [audio_io_base]
0001556C 6683C204               <1>       add    dx, 4 ; Mixer chip address port
00015570 B022                   <1>       mov    al, 22h ; master volume
00015572 EE                     <1>       out    dx, al
00015573 6642                   <1>       inc    dx
00015575 8A25[CE9C0100]         <1>       mov    ah, [audio_master_volume_l]
0001557B C0EC02                 <1>       shr    ah, 2 ; 32 -> 8 level
0001557E C0E405                 <1>       shl    ah, 5 ; bit 5 to 7
00015581 A0[CF9C0100]           <1>       mov    al, [audio_master_volume_r]
00015586 C0E802                 <1>       shr    al, 2 ; 32 -> 8 level
                                <1>       ;and   al, 0Fh
00015589 D0E0                   <1>       shl    al, 1 ; bit 1 to 3
0001558B 08E0                   <1>       or     al, ah
0001558D EE                     <1>       out    dx, al
0001558E 665A                   <1>       pop    dx ; DX (port address) must be restored
00015590 C3                     <1>       retn
                                <1>
                                <1> sb16_pause:
00015591 668B15[9E9C0100]       <1>       mov    dx, [audio_io_base]
00015598 6683C20C               <1>       add    dx, 0Ch ; Command & Data Port
0001559C 803D[C89C0100]10       <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
000155A3 7404                   <1>       je     short sb_pause_1
                                <1>       ; 8 bit samples
000155A5 B3D0                   <1>       mov    bl, 0D0h ; 8 bit DMA mode
000155A7 EB02                   <1>       jmp    short sb_pause_2
                                <1> sb_pause_1:
                                <1>       ; 16 bit samples
000155A9 B3D5                   <1>       mov    bl, 0D5h ; 16 bit DMA mode
                                <1> sb_pause_2:
                                <1>       SbOut   bl ; bCommand
                                <2> %%Wait:
000155AB EC                     <2>  in al, dx
000155AC 08C0                   <2>  or al, al
000155AE 78FB                   <2>  js short %%Wait
000155B0 88D8                   <2>  mov al, %1
000155B2 EE                     <2>  out dx, al
                                <1> sb_pause_3:
000155B3 C3                     <1>       retn
                                <1>
                                <1> sb16_continue:
000155B4 668B15[9E9C0100]       <1>       mov    dx, [audio_io_base]
000155BB 6683C20C               <1>       add    dx, 0Ch ; Command & Data Port
000155BF 803D[C89C0100]10       <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
000155C6 7404                   <1>       je     short sb_cont_1
                                <1>       ; 8 bit samples
000155C8 B3D4                   <1>       mov    bl, 0D4h ; 8 bit DMA mode
000155CA EB02                   <1>       jmp    short sb_cont_2
                                <1> sb_cont_1:
                                <1>       ; 16 bit samples
000155CC B3D6                   <1>       mov    bl, 0D6h ; 16 bit DMA mode
                                <1> sb_cont_2:
                                <1>       SbOut   bl ; bCommand
                                <2> %%Wait:
000155CE EC                     <2>  in al, dx
000155CF 08C0                   <2>  or al, al
000155D1 78FB                   <2>  js short %%Wait
000155D3 88D8                   <2>  mov al, %1
000155D5 EE                     <2>  out dx, al
                                <1> sb_cont_3:
000155D6 C3                     <1>       retn
                                <1>
                                <1> sb16_stop:
                                <1>       ; 24/04/2017
000155D7 803D[CC9C0100]00       <1>       cmp    byte [audio_play_cmd], 0
000155DE 7648                   <1>       jna    short sb16_stop_4
                                <1>
                                <1>       ; 22/05/2017
000155E0 668B15[9E9C0100]       <1>       mov    dx, [audio_io_base]
000155E7 6683C20C               <1>       add    dx, 0Ch
                                <1>
000155EB B3D9                   <1>       mov    bl, 0D9h ; exit auto-initialize 16 bit transfer
                                <1>       ; stop  autoinitialized DMA transfer mode
000155ED 803D[C89C0100]10       <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
000155F4 7402                   <1>       je     short sb16_stop_1
                                <1>       ;mov   bl, 0DAh ; exit auto-initialize 8 bit transfer
000155F6 FEC3                   <1>       inc    bl
                                <1> sb16_stop_1:
                                <1>       SbOut  bl ; exit auto-initialize transfer command
                                <2> %%Wait:
000155F8 EC                     <2>  in al, dx
000155F9 08C0                   <2>  or al, al
000155FB 78FB                   <2>  js short %%Wait
000155FD 88D8                   <2>  mov al, %1
000155FF EE                     <2>  out dx, al
                                <1>
00015600 30C0                   <1>       xor     al, al ; stops all DMA processes on selected channel
                                <1>
00015602 803D[C89C0100]10       <1>       cmp    byte [audio_bps], 16 ; 16 bit samples
00015609 7404                   <1>       je     short sb16_stop_2
```

```
1917 0001560B E60C               <1>       out    0Ch, al ; clear selected channel register
1918 0001560D EB02               <1>       jmp    short sb16_stop_3
1919                             <1>
1920                             <1> sb16_stop_2:
1921 0001560F E6D8               <1>       out    0D8h, al ; clear selected channel register
1922                             <1>
1923                             <1> sb16_stop_3:
1924 00015611 C605[CC9C0100]00   <1>       mov    byte [audio_play_cmd], 0 ; stop !
1925                             <1> SbDone:
1926                             <1>       ;mov   dx, [audio_io_base]
1927                             <1>       ;add   dx, 0Ch
1928                             <1>       SbOut   0D0h
1928                             <2> %%Wait:
1928 00015618 EC                 <2>  in al, dx
1928 00015619 08C0               <2>  or al, al
1928 0001561B 78FB               <2>  js short %%Wait
1928 0001561D B0D0               <2>  mov al, %1
1928 0001561F EE                 <2>  out dx, al
1929                             <1>       SbOut   0D3h
1929                             <2> %%Wait:
1929 00015620 EC                 <2>  in al, dx
1929 00015621 08C0               <2>  or al, al
1929 00015623 78FB               <2>  js short %%Wait
1929 00015625 B0D3               <2>  mov al, %1
1929 00015627 EE                 <2>  out dx, al
1930                             <1> sb16_stop_4:
1931 00015628 C3                 <1>       retn
1932                             <1>
1933                             <1> sb16_reset:
1934                             <1>       ; 24/04/2017
1935 00015629 668B15[9E9C0100]   <1>       mov     dx, [audio_io_base] ; try to reset the DSP.
1936 00015630 6683C206           <1>       add     dx, 06h
1937 00015634 B001               <1>       mov    al, 1
1938 00015636 EE                 <1>       out    dx, al
1939                             <1>
1940 00015637 EC                 <1>       in     al, dx
1941 00015638 EC                 <1>       in     al, dx
1942 00015639 EC                 <1>       in     al, dx
1943 0001563A EC                 <1>       in     al, dx
1944                             <1>
1945 0001563B 30C0               <1>       xor     al, al
1946 0001563D EE                 <1>       out    dx, al
1947                             <1>
1948 0001563E 6683C208           <1>       add     dx, 08h
1949 00015642 66B96400           <1>       mov    cx, 100
1950                             <1> sbrstWaitID:
1951 00015646 EC                 <1>       in     al, dx
1952 00015647 08C0               <1>       or      al, al
1953 00015649 7804               <1>       js      short sbrstGetID
1954 0001564B E2F9               <1>       loop    sbrstWaitID
1955 0001564D F9                 <1>       stc
1956 0001564E C3                 <1>       retn
1957                             <1> sbrstGetID:
1958 0001564F 6683EA04           <1>       sub     dx, 04h
1959 00015653 EC                 <1>       in     al, dx
1960 00015654 3CAA               <1>       cmp     al, 0AAh
1961 00015656 7406               <1>       je      short sb_rst_retn
1962 00015658 6683C204           <1>       add     dx, 04h
1963 0001565C E2E8               <1>       loop    sbrstWaitID
1964                             <1> sb_rst_retn:
1965 0001565E C3                 <1>       retn
1966                             <1>
1967                             <1> ac97_codec_config:
1968                             <1>       ; 10/06/2017
1969                             <1>       ; 05/06/2017
1970                             <1>       ; 29/05/2017
1971                             <1>       ; 28/05/2017 (TRDOS 386, 'audio.s')
1972                             <1>       ; 07/11/2016 (Erdogan Tan)
1973                             <1>       ; Derived from 'codecConfig' procedure in 'CODEC.ASM'
1974                             <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
1975                             <1>
1976                             <1>       ;; 'PLAYER.ASM'
1977                             <1>       ;; get ICH base address regs for mixer and bus master
1978                             <1>
1979                             <1> init_ac97_controller: ; 10/06/2017
1980 0001565F A1[A09C0100]       <1>       mov    eax, [audio_dev_id]
1981                             <1>         ;mov al, NAMBAR_REG
1982                             <1>         ;;call  pciRegRead16                ; read PCI registers 10-11
1983                             <1>         ;call      pciRegRead32
1984                             <1>       ;and   dx, IO_ADDR_MASK          ; mask off BIT0
1985                             <1>       ;;and  edx, IO_ADDR_MASK
1986                             <1>
1987                             <1>         ;mov [NAMBAR], dx              ; save audio mixer base addr
1988                             <1>
1989                             <1>         ;mov    al, NABMBAR_REG
1990                             <1>         ;;call     pciRegRead16
1991                             <1>         ;call      pciRegRead32
1992                             <1>       ;and   dx, 0FFC0h ; IO_ADDR_MASK
1993                             <1>       ;;and  edx, 0FFC0h
1994                             <1>
1995                             <1>         ;mov    [NABMBAR], dx               ; save bus master base addr
1996                             <1>
1997                             <1>       ;mov   eax, [audio_dev_id]
1998 00015664 B004               <1>         mov     al, PCI_CMD_REG
1999                             <1>         ;call      pciRegRead8              ; read PCI command register
2000 00015666 E84AF8FFFF         <1>         call pciRegRead16
2001 0001566B 80CA05             <1>       or      dl, IO_ENA+BM_ENA              ; enable IO and bus master
2002                             <1>         ;call      pciRegWrite8
2003 0001566E E8ADF8FFFF         <1>       call   pciRegWrite16
2004                             <1>
2005                             <1>       ; 'CODEC.ASM'
2006                             <1>
2007                             <1>       ; enable codec, unmute stuff, set output rate
2008                             <1> ;     ; entry: [audio_freq] = desired sample rate
2009                             <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>
00015673 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>         ;
00015678 66BA2C00               <1>         mov     dx, GLOB_CNT_REG ; 2Ch
0001567C 660315[D29C0100]       <1>         add     dx, [NABMBAR]
00015683 ED                     <1>         in      eax, dx
                                <1>         ; ?
00015684 66BA3000               <1>         mov     dx, GLOB_STS_REG ; 30h
00015688 660315[D29C0100]       <1>         add     dx, [NABMBAR]
0001568F ED                     <1>         in      eax, dx
                                <1>
00015690 83F8FF                 <1>         cmp     eax, 0FFFFFFFFh ; -1
00015693 744B                   <1>         je      short init_ac97_codec_err1
                                <1>
00015695 A900030010             <1>         test    eax, CTRL_ST_CREADY
0001569A 7507                   <1>         jnz     short _ac97_codec_ready
                                <1>
0001569C E8EF020000             <1>         call    reset_ac97_codec
000156A1 723E                   <1>         jc      short init_ac97_codec_err2
                                <1>
                                <1> _ac97_codec_ready:
000156A3 668B15[D09C0100]       <1>         mov     dx, [NAMBAR]
                                <1>         ;add    dx, 0 ; ac_reg_0 ; reset register
000156AA 66EF                   <1>         out     dx, ax
                                <1>
000156AC 31C0                   <1>         xor     eax, eax ; 0
000156AE 668B15[D09C0100]       <1>         mov     dx, [NAMBAR]
000156B5 6683C226               <1>         add     dx, CODEC_REG_POWERDOWN
000156B9 66EF                   <1>         out     dx, ax
                                <1>
                                <1>         ; 10/06/2017
                                <1>         ; 29/05/2017
                                <1>         ; wait for 1 second
000156BB B9E8030000             <1>         mov     ecx, 1000 ; 1000*0.25ms = 1s
                                <1> _ac97_codec_rloop:
000156C0 E807F9FFFF             <1>         call    delay1_4ms
000156C5 E802F9FFFF             <1>         call    delay1_4ms
000156CA E8FDF8FFFF             <1>         call    delay1_4ms
000156CF E8F8F8FFFF             <1>         call    delay1_4ms
                                <1>         ;mov    dx, [NAMBAR]
                                <1>         ;add    dx, CODEC_REG_POWERDOWN
000156D4 66ED                   <1>         in      ax, dx
000156D6 6683E00F               <1>         and     ax, 0Fh
000156DA 3C0F                   <1>         cmp     al, 0Fh
000156DC 7404                   <1>         je      short _ac97_codec_init_ok
000156DE E2E0                   <1>         loop    _ac97_codec_rloop
                                <1>
                                <1> init_ac97_codec_err1:
000156E0 F9                     <1>         stc
                                <1> init_ac97_codec_err2:
000156E1 C3                     <1>         retn
                                <1>
                                <1> _ac97_codec_init_ok:
000156E2 B002                   <1>           mov     al, 2 ; force set 16-bit 2-channel PCM
000156E4 66BA2C00               <1>         mov     dx, GLOB_CNT_REG ; 2Ch
000156E8 660315[D29C0100]       <1>         add     dx, [NABMBAR]
000156EF EF                     <1>         out     dx, eax
                                <1>
                                <1>         ;call  delay1_4ms
                                <1>
                                <1>         ; 10/06/2017
000156F0 E849020000             <1>         call    reset_ac97_controller
                                <1>
                                <1> ;       call    setup_ac97_codec
                                <1> ;
                                <1> ;detect_ac97_codec:
                                <1> ;       retn
                                <1>
                                <1> setup_ac97_codec:
```

```
                                <1>      ; 22/07/2020
                                <1>      ; 10/06/2017
                                <1>      ; 29/05/2017
000156F5 B802020000             <1>      mov     eax, 0202h
000156FA 66A3[CE9C0100]         <1>      mov    [audio_master_volume], ax
00015700 66B81F1F               <1>      mov    ax, 1F1Fh ; 31, 31
                                <1>
00015704 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
0001570B 6683C202               <1>      add     dx, CODEC_MASTER_VOL_REG ;02h
0001570F 6631C0                 <1>      xor    ax, ax       ; volume attenuation = 0 (max. volume)
00015712 66EF                   <1>      out     dx, ax
                                <1>
00015714 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
0001571B 6683C206               <1>      add     dx, CODEC_MASTER_MONO_VOL_REG  ;06h
                                <1>      ;xor   ax, ax
0001571F 66EF                   <1>      out     dx, ax
                                <1>
00015721 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
00015728 6683C20A               <1>      add     dx, CODEC_PCBEEP_VOL_REG ;0Ah
                                <1>      ;xor    ax, ax
0001572C 66EF                   <1>      out     dx, ax
                                <1>
0001572E 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
00015735 6683C218               <1>      add     dx, CODEC_PCM_OUT_REG          ;18h
                                <1>      ;xor    ax, ax
00015739 66EF                   <1>      out     dx, ax
                                <1>
0001573B 66B80880               <1>      mov     ax, 8008h ; Mute
0001573F 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
                                <1>      ; 22/07/2020
00015746 6683C20C               <1>      add    dx, CODEC_PHONE_VOL_REG          ;0Ch
                                <1>                             ; AC97_PHONE_VOL ; TAD Input (Mono)
0001574A 66EF                   <1>      out     dx, ax
                                <1>
0001574C 66B80808               <1>        mov  ax, 0808h
00015750 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
00015757 6683C210               <1>        add  dx, CODEC_LINE_IN_VOL_REG ;10h ; Line Input (Stereo)
0001575B 66EF                   <1>      out     dx, ax
                                <1>
                                <1>      ;mov   ax, 0808h
0001575D 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
00015764 6683C212               <1>        add  dx, CODEC_CD_VOL_REG ;12h ; CR Input (Stereo)
00015768 66EF                   <1>      out     dx, ax
                                <1>
                                <1>      ;mov   ax, 0808h
0001576A 668B15[D09C0100]       <1>      mov     dx, [NAMBAR]
00015771 6683C216               <1>        add  dx, CODEC_AUX_VOL_REG ;16h ; Aux Input (Stereo)
00015775 66EF                   <1>      out     dx, ax
                                <1>
                                <1>        ;call    delay1_4ms
                                <1>        ;call    delay1_4ms
                                <1>        ;call    delay1_4ms
                                <1>        ;call    delay1_4ms
                                <1>
                                <1> detect_ac97_codec:
00015777 C3                     <1>        retn
                                <1>
                                <1> set_ac97_bdl: ; Set AC97 (ICH) Buffer Descriptor List
                                <1>      ; 17/06/2017
                                <1>      ; 11/06/2017
                                <1>      ; 28/05/2017
                                <1>      ; eax = dma buffer address = [audio_DMA_buff]
                                <1>      ; ecx = dma buffer buffer size = [audio_dmabuff_size]
                                <1>
00015778 D1E9                   <1>      shr    ecx, 1 ; dma half buffer size
0001577A 89CE                   <1>      mov    esi, ecx
                                <1>
0001577C BF[D49C0100]           <1>        mov     edi, audio_bdl_buff    ; get BDL address
00015781 B910000000             <1>        mov     ecx, 32 / 2            ; make 32 entries in BDL
                                <1>
00015786 EB05                   <1>      jmp    short s_ac97_bdl1
                                <1>
                                <1> s_ac97_bdl0:
                                <1>      ; set buffer descriptor 0 to start of data file in memory
                                <1>
00015788 A1[B89C0100]           <1>      mov    eax, [audio_dma_buff]     ; Physical address of DMA buffer
                                <1>
                                <1> s_ac97_bdl1:
0001578D AB                     <1>      stosd                     ; store dmabuffer1 address
                                <1>
0001578E 89C2                   <1>      mov    edx, eax
                                <1>
                                <1> ;
                                <1> ; Buffer Descriptors List
                                <1> ; As stated earlier, each buffer descriptor list is a set of (up to) 32
                                <1> ; descriptors, each 8 bytes in length. Bytes 0-3 of a descriptor entry point
                                <1> ; to a chunk of memory to either play from or record to. Bytes 4-7 of an
                                <1> ; entry describe various control things detailed below.
                                <1> ;
                                <1> ; Buffer pointers must always be aligned on a Dword boundry.
                                <1> ;
                                <1> ;
                                <1>
                                <1> ;IOC                     equ     BIT31 ; Fire an interrupt whenever this
                                <1>                                        ; buffer is complete.
                                <1>
                                <1> ;BUP                     equ     BIT30  ; Buffer Underrun Policy.
                                <1>                                         ; if this buffer is the last buffer
                                <1>                                         ; in a playback, fill the remaining
                                <1>                                         ; samples with 0 (silence) or not.
                                <1>                                         ; It's a good idea to set this to 1
                                <1>                                         ; for the last buffer in playback,
                                <1>                                         ; otherwise you're likely to get a lot
                                <1>                                         ; of noise at the end of the sound.
                                <1>
```

```
                        <1> ;
                        <1> ; Bits 15:0 contain the length of the buffer, in number of samples, which
                        <1> ; are 16 bits each, coupled in left and right pairs, or 32bits each.
                        <1> ; Luckily for us, that's the same format as .wav files.
                        <1> ;
                        <1> ; A value of FFFF is 65536 samples. Running at 44.1Khz, that's just about
                        <1> ; 1.5 seconds of sample time. FFFF * 32bits is 1FFFFh bytes or 128k of data.
                        <1> ;
                        <1> ; A value of 0 in these bits means play no samples.
                        <1> ;
                        <1>
00015790 89F0           <1>       mov    eax, esi ; DMA half buffer size
00015792 01C2           <1>       add    edx, eax
00015794 D1E8           <1>       shr    eax, 1 ; count of 16 bit samples
                        <1>       ;or    eax, IOC+BUP
00015796 0D00000080     <1>       or     eax, IOC ; 11/06/2017
0001579B AB             <1>       stosd
                        <1>
                        <1> ; 2nd buffer:
                        <1>
0001579C 89D0           <1>         mov  eax, edx ; Physical address of the 2nd half of DMA buffer
0001579E 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> ;
0001579F 89F0           <1>       mov    eax, esi ; DMA half buffer size
000157A1 D1E8           <1>       shr    eax, 1 ; count of 16 bit samples
                        <1>       ;or    eax, IOC+BUP
000157A3 0D00000080     <1>       or     eax, IOC ; 11/06/2017
000157A8 AB             <1>       stosd
                        <1>
000157A9 E2DD           <1>       loop    s_ac97_bd10
                        <1>
000157AB 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>
000157AC 668B15[D09C0100]   <1>       mov          dx, [NAMBAR]
000157B3 6683C22A           <1>       add          dx, CODEC_EXT_AUDIO_CTRL_REG     ; 2Ah
000157B7 66ED               <1>       in           ax, dx
000157B9 6683C801           <1>       or     ax, 1
000157BD 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>
000157BF 66A1[CA9C0100]     <1>       mov    ax, [audio_freq]          ; sample rate
                        <1>
000157C5 668B15[D09C0100]   <1>       mov          dx, [NAMBAR]
000157CC 6683C22C           <1>       add          dx, CODEC_PCM_FRONT_DACRATE_REG  ; 2Ch
000157D0 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> ;
000157D2 668B15[D29C0100]   <1>         mov     dx, [NABMBAR]
000157D9 6683C21B           <1>         add     dx, PO_CR_REG                   ; set pointer to Cntl reg
000157DD B002               <1>         mov     al, RR                          ; set reset
000157DF 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> ;
000157E0 B8[D49C0100]       <1>         mov  eax, audio_bdl_buff
000157E5 668B15[D29C0100]   <1>       mov    dx, [NABMBAR]
000157EC 6683C210           <1>       add    dx, PO_BDBAR_REG
```

```
2325 000157F0 EF                  <1>       out    dx, eax
2326                              <1> ;
2327                              <1> ; All set. Let's play some music.
2328                              <1> ;
2329                              <1> ;
2330 000157F1 B81F000000          <1>       mov    eax, 31
2331 000157F6 E816000000          <1>       call    set_ac97_LastValidIndex
2332                              <1>
2333 000157FB C605[CC9C0100]01    <1>       mov    byte [audio_play_cmd], 1 ; play command (do not stop) !
2334                              <1>
2335                              <1> ac97_play: ; continue to play (after pause)
2336                              <1>       ; 11/06/2017
2337                              <1>       ; 29/05/2017
2338                              <1>       ; 28/05/2017
2339 00015802 668B15[D29C0100]    <1>         mov     dx, [NABMBAR]
2340 00015809 6683C21B            <1>         add     dx, PO_CR_REG          ; PCM out control register
2341 0001580D B011                <1>         mov  al, IOCE+RPBM ; 29/05/2017
2342                              <1>         ;mov al, 1Dh ; (Ref: KolibriOS, intelac97.asm, 'play:')
2343 0001580F EE                  <1>       out     dx, al                   ; set start!
2344                              <1>
2345                              <1>       ;mov   byte [audio_play_cmd], 1 ; play command (do not stop) !
2346                              <1>
2347 00015810 C3                  <1>       retn
2348                              <1>
2349                              <1> ;input AL = index # to stop on
2350                              <1> set_ac97_LastValidIndex:
2351                              <1>       ; 28/05/2017
2352                              <1>       ; Derived from 'setLastValidIndex' procedure in 'ICHWAV.ASM'
2353                              <1>       ; .wav player for DOS by Jeff Leyda (02/09/2002)
2354 00015811 668B15[D29C0100]    <1>       mov    dx, [NABMBAR]
2355 00015818 6683C215            <1>       add    dx, PO_LVI_REG
2356 0001581C EE                  <1>         out     dx, al
2357                              <1>       ;mov   [audio_lvi], al ; for ac97 int handler
2358 0001581D C3                  <1>       retn
2359                              <1>
2360                              <1> ac97_volume:
2361                              <1>       ; 28/05/2017
2362                              <1>       ; bl = component (0 = master/playback/lineout volume)
2363                              <1>       ; cl = left channel volume level (0 to 31)
2364                              <1>       ; ch = right channel volume level (0 to 31)
2365                              <1>
2366 0001581E 08DB                <1>       or     bl, bl
2367 00015820 7523                <1>       jnz    short ac97_vol_1 ; temporary !
2368 00015822 66B81F1F            <1>       mov    ax, 1F1Fh ; 31,31
2369 00015826 38C1                <1>       cmp    cl, al
2370 00015828 771B                <1>       ja     short ac97_vol_1 ; temporary !
2371 0001582A 38E5                <1>       cmp    ch, ah
2372 0001582C 7717                <1>       ja     short ac97_vol_1 ; temporary !
2373 0001582E 66890D[CE9C0100]    <1>       mov    [audio_master_volume], cx
2374 00015835 6629C8              <1>       sub    ax, cx
2375 00015838 668B15[D09C0100]    <1>       mov     dx, [NAMBAR]
2376 0001583F 6683C202            <1>       add     dx, CODEC_MASTER_VOL_REG  ; 02h ; Line Out
2377 00015843 66EF                <1>       out     dx, ax
2378                              <1> ac97_vol_1:
2379 00015845 C3                  <1>       retn
2380                              <1>
2381                              <1> ac97_int_handler:
2382                              <1>       ; 12/10/2017
2383                              <1>       ; 10/10/2017
2384                              <1>       ; 09/10/2017
2385                              <1>       ; 13/06/2017, 13/06/2017
2386                              <1>       ; 10/06/2017, 11/06/2017
2387                              <1>       ; Interrupt Handler for AC97 (ICH) Audio Controller
2388                              <1>       ; Note: called by 'dev_IRQ_service'
2389                              <1>       ; 28/05/2017
2390                              <1>
2391                              <1>       ;push  eax ; * must be saved !
2392                              <1>       ;push  edx
2393                              <1>       ;push  ecx
2394                              <1>       ;push  ebx ; * must be saved !
2395                              <1>       ;push  esi
2396                              <1>       ;push  edi
2397                              <1>
2398                              <1>       ;cmp   byte [audio_busy], 1
2399                              <1>       ;jnb   _ac97_ih2    ; busy !
2400                              <1>
2401 00015846 66BA3000            <1>         mov     dx, GLOB_STS_REG
2402 0001584A 660315[D29C0100]    <1>         add  dx, [NABMBAR]
2403 00015851 ED                  <1>       in     eax, dx
2404                              <1>
2405 00015852 83F8FF              <1>         cmp     eax, 0FFFFFFFFh ; -1
2406 00015855 0F849A000000        <1>         je      _ac97_ih3 ; exit
2407                              <1>
2408 0001585B A940000000          <1>         test    eax, 40h ; PCM Out Interrupt
2409 00015860 750E                <1>         jnz     short _ac97_ih0
2410                              <1>
2411 00015862 85C0                <1>       test   eax, eax
2412 00015864 0F848B000000        <1>       jz     _ac97_ih3 ; exit
2413                              <1>
2414                              <1>       ;mov   dx, GLOB_STS_REG
2415                              <1>         ;add dx, [NABMBAR]
2416 0001586A EF                  <1>       out    dx, eax
2417                              <1>
2418 0001586B E985000000          <1>       jmp    _ac97_ih3 ; exit
2419                              <1>
2420                              <1> _ac97_ih0:
2421 00015870 50                  <1>       push   eax
2422                              <1>       ; 09/10/2017
2423 00015871 803D[CC9C0100]01    <1>       cmp    byte [audio_play_cmd], 1
2424 00015878 727C                <1>       jb     short _ac97_ih4 ; stop command !
2425                              <1>
2426                              <1>       ;mov   byte [audio_busy], 1
2427                              <1>
2428                              <1>       ;mov   al, 10h
2429                              <1>       ;mov   dx, PO_CR_REG
```

```
                                <1>         ;add dx, [NABMBAR]
                                <1>       ;out   dx, al
                                <1>
0001587A 66B81C00               <1>       mov    ax, 1Ch ; FIFOE(=16)+BCIS(=8)+LVBCI(=4)
0001587E 66BA1600               <1>       mov    dx, PO_SR_REG
00015882 660315[D29C0100]       <1>         add  dx, [NABMBAR]
00015889 66EF                   <1>       out    dx, ax
                                <1>
0001588B 66BA1400               <1>       mov    dx, PO_CIV_REG
0001588F 660315[D29C0100]       <1>         add  dx, [NABMBAR]
00015896 EC                     <1>       in     al, dx
                                <1>
                                <1>       ;cmp   al, [audio_civ] ; [audio_flag]
                                <1>       ;je    short _ac97_ih2
                                <1>
00015897 A2[CD9C0100]           <1>       mov    [audio_civ], al
0001589C FEC8                   <1>       dec    al
                                <1>       ;inc   al ; 11/06/2017
0001589E 241F                   <1>       and    al, 1Fh
                                <1>
000158A0 66BA1500               <1>         mov     dx, PO_LVI_REG
000158A4 660315[D29C0100]       <1>         add  dx, [NABMBAR]
000158AB EE                     <1>         out  dx, al
                                <1>
                                <1>       ; 12/10/2017
000158AC A0[CD9C0100]           <1>       mov    al, [audio_civ]
000158B1 FEC0                   <1>       inc    al
000158B3 2401                   <1>       and    al, 1
000158B5 A2[C09C0100]           <1>       mov    [audio_flag], al
                                <1>       ;; [audio_flag] : 0 = Buffer 1, 1 = Buffer 2
                                <1>       ;
000158BA 58                     <1>       pop    eax
                                <1>       ;
000158BB 83E040                 <1>       and     eax, 40h
000158BE 668B15[D29C0100]       <1>         mov  dx, [NABMBAR]
000158C5 6683C230               <1>       add    dx, GLOB_STS_REG
000158C9 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
000158CA 8B3D[B89C0100]         <1>       mov    edi, [audio_dma_buff]
000158D0 8B0D[BC9C0100]         <1>       mov    ecx, [audio_dmabuff_size]
000158D6 D1E9                   <1>       shr    ecx, 1 ; dma buff size / 2 = half buffer size
                                <1>
                                <1>       ; 12/10/2017
000158D8 803D[C09C0100]00       <1>       cmp    byte [audio_flag], 0
000158DF 7702                   <1>       ja     short _ac97_ih1  ; Playing Half Buffer 2 (Current: FLAG)
                                <1>       ; Playing Half Buffer 1 (Current: EOL)
000158E1 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>
000158E3 8B35[B09C0100]         <1>       mov    esi, [audio_p_buffer] ; phy addr of audio buff
000158E9 C1E902                 <1>       shr    ecx, 2 ; half buff size / 4
000158EC F3A5                   <1>       rep    movsd
                                <1>
                                <1>       ; 10/10/2017
                                <1>       ; switch flag value
000158EE 8035[C09C0100]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>
000158F5 C3                     <1>       retn
                                <1>
                                <1> _ac97_ih4:
                                <1>       ; 09/10/2017
000158F6 E818000000             <1>       call   _ac97_stop
                                <1>       ;
000158FB 58                     <1>       pop    eax
                                <1>       ;
000158FC 83E040                 <1>       and     eax, 40h
000158FF 668B15[D29C0100]       <1>         mov  dx, [NABMBAR]
00015906 6683C230               <1>       add    dx, GLOB_STS_REG
0001590A 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
0001590B C3                     <1>       retn
                                <1>
```

```
                            <1> ac97_stop:
                            <1>       ; 28/05/2017
0001590C C605[CC9C0100]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
00015913 30C0               <1>       xor   al, al ; 0
00015915 E813000000         <1>       call  ac97_po_cmd
                            <1>
                            <1>       ; (Ref: KolibriOS, intelac97.asm, 'stop:')
                            <1>       ; Clear FIFOE, BCIS, LVBCI (Ref: Intel ICH hub manual)
0001591A 66B81C00           <1>       mov     ax, 1Ch
0001591E 668B15[D29C0100]   <1>       mov     dx, [NABMBAR]
00015925 6683C216           <1>       add     dx, PO_SR_REG
00015929 66EF               <1>       out     dx, ax
                            <1>
                            <1>       ;retn
                            <1>
                            <1>       ; 11/06/2017
0001592B B002               <1>       mov     al, RR
                            <1> ac97_po_cmd:
                            <1>        ;11/06/2017
                            <1>       ; 29/05/2017
0001592D 668B15[D29C0100]   <1>       mov     dx, [NABMBAR]
00015934 6683C21B           <1>         add     dx, PO_CR_REG         ; PCM out control register
00015938 EE                 <1>       out   dx, al
00015939 C3                 <1>       retn
                            <1>
                            <1> ac97_pause:
                            <1>       ; 11/06/2017
                            <1>       ; 29/05/2017
0001593A B010               <1>       mov   al, IOCE
0001593C 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
0001593E 31C0               <1>       xor     eax, eax
00015940 66BA0B00           <1>         mov  dx, PI_CR_REG
00015944 660315[D29C0100]   <1>       add   dx, [NABMBAR]
0001594B EE                 <1>       out     dx, al
                            <1>
0001594C 66BA1B00           <1>         mov     dx, PO_CR_REG
00015950 660315[D29C0100]   <1>       add   dx, [NABMBAR]
00015957 EE                 <1>       out     dx, al
                            <1>
00015958 66BA2B00           <1>         mov     dx, MC_CR_REG
0001595C 660315[D29C0100]   <1>       add   dx, [NABMBAR]
00015963 EE                 <1>       out     dx, al
                            <1>
00015964 B002               <1>         mov     al, RR
00015966 66BA0B00           <1>         mov     dx, PI_CR_REG
0001596A 660315[D29C0100]   <1>       add   dx, [NABMBAR]
00015971 EE                 <1>       out     dx, al
                            <1>
00015972 66BA1B00           <1>         mov     dx, PO_CR_REG
00015976 660315[D29C0100]   <1>       add   dx, [NABMBAR]
0001597D EE                 <1>       out     dx, al
                            <1>
0001597E 66BA2B00           <1>         mov     dx, MC_CR_REG
00015982 660315[D29C0100]   <1>       add   dx, [NABMBAR]
00015989 EE                 <1>       out     dx, al
                            <1>
0001598A C3                 <1>       retn
                            <1>
                            <1> ac97_reset:
                            <1>       ; 10/06/2017
                            <1>       ; 29/05/2017
                            <1>       ; 28/05/2017
0001598B 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)
00015990 66BA2C00           <1>       mov   dx, GLOB_CNT_REG ; 2Ch
00015994 660315[D29C0100]   <1>       add   dx, [NABMBAR]
0001599B ED                 <1>       in    eax, dx
                            <1>
0001599C A902000000         <1>       test  eax, 2
000159A1 7407               <1>       jz    short _r_ac97codec_cold
                            <1>
000159A3 E80F000000         <1>       call  warm_ac97codec_reset
000159A8 7308               <1>       jnc   short _r_ac97codec_ok
                            <1> _r_ac97codec_cold:
000159AA E83D000000         <1>         call    cold_ac97codec_reset
000159AF 7301               <1>         jnc     short _r_ac97codec_ok
                            <1>
                            <1>       ; 16/04/2017
                            <1>         ;xor     eax, eax         ; timeout error
                            <1>             ;stc
000159B1 C3                 <1>       retn
                            <1>
                            <1> _r_ac97codec_ok:
000159B2 31C0               <1>         xor     eax, eax
                            <1>         ;mov al, VIA_ACLINK_C00_READY ; 1
000159B4 FEC0               <1>         inc al
000159B6 C3                 <1>       retn
                            <1>
```

```
                            <1> warm_ac97codec_reset:
                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
000159B7 B806000000         <1>         mov     eax, 6
000159BC 66BA2C00           <1>       mov     dx, GLOB_CNT_REG ; 2Ch
000159C0 660315[D29C0100]   <1>       add     dx, [NABMBAR]
000159C7 EF                 <1>       out     dx, eax
                            <1>
000159C8 B90A000000         <1>       mov     ecx, 10       ; total 1s
                            <1> _warm_ac97c_rst_wait:
000159CD 51                 <1>       push    ecx
000159CE E8ECF5FFFF         <1>       call    delay_100ms
000159D3 59                 <1>       pop     ecx
                            <1>
000159D4 66BA3000           <1>       mov     dx, GLOB_STS_REG ; 30h
000159D8 660315[D29C0100]   <1>       add     dx, [NABMBAR]
000159DF ED                 <1>       in      eax, dx
                            <1>
000159E0 A900030010         <1>       test    eax, CTRL_ST_CREADY
000159E5 7504               <1>       jnz     short _warm_ac97c_rst_ok
                            <1>
000159E7 49                 <1>         dec     ecx
000159E8 75E3               <1>         jnz     short _warm_ac97c_rst_wait
                            <1>
                            <1> _warm_ac97c_rst_fail:
000159EA F9                 <1>         stc
                            <1> _warm_ac97c_rst_ok:
000159EB C3                 <1>       retn
                            <1>
                            <1> cold_ac97codec_reset:
                            <1>       ; 28/05/2017 - Erdogan Tan (Ref: KolibriOS, intelac97.asm)
000159EC B802000000         <1>         mov     eax, 2
000159F1 66BA2C00           <1>       mov     dx, GLOB_CNT_REG ; 2Ch
000159F5 660315[D29C0100]   <1>       add     dx, [NABMBAR]
000159FC EF                 <1>       out     dx, eax
                            <1>
000159FD E8BDF5FFFF         <1>       call    delay_100ms  ; wait 100 ms
00015A02 E8B8F5FFFF         <1>       call    delay_100ms  ; wait 100 ms
00015A07 E8B3F5FFFF         <1>       call    delay_100ms  ; wait 100 ms
00015A0C E8AEF5FFFF         <1>       call    delay_100ms  ; wait 100 ms
                            <1>
00015A11 B910000000         <1>       mov     ecx, 16       ; total 20*100 ms = 2s
                            <1> _cold_ac97c_rst_wait:
00015A16 66BA3000           <1>       mov     dx, GLOB_STS_REG ; 30h
00015A1A 660315[D29C0100]   <1>       add     dx, [NABMBAR]
00015A21 ED                 <1>       in      eax, dx
                            <1>
00015A22 A900030010         <1>       test    eax, CTRL_ST_CREADY
00015A27 750B               <1>       jnz     short _cold_ac97c_rst_ok
                            <1>
00015A29 51                 <1>       push    ecx
00015A2A E890F5FFFF         <1>       call    delay_100ms
00015A2F 59                 <1>       pop     ecx
                            <1>
00015A30 49                 <1>         dec     ecx
00015A31 75E3               <1>         jnz     short _cold_ac97c_rst_wait
                            <1>
                            <1> _cold_ac97c_rst_fail:
00015A33 F9                 <1>         stc
                            <1> _cold_ac97c_rst_ok:
00015A34 C3                 <1>       retn
                            <1>
                            <1> sb16_current_sound_data:
                            <1>       ; 20/08/2017
                            <1>       ; 24/06/2017
                            <1>       ; 22/06/2017
                            <1>       ; get current sound (PCM out) data for graphics
                            <1>       ; (for Sound Blaster 16)
                            <1>       ; ebx = Physical address (on page boundary)
                            <1>       ; ecx = Byte count
                            <1>       ; [audio_buff_size]
                            <1>
                            <1>       ;;mov  edi, [audio_buff_size]
                            <1>       ;mov   edi, [audio_dmabuff_size]
                            <1>       ;mov   esi, [audio_dma_buff]
00015A35 39CF               <1>       cmp    edi, ecx
00015A37 7302               <1>       jnb    short sb16_gcd_0
00015A39 89F9               <1>       mov    ecx, edi
                            <1> sb16_gcd_0:
                            <1>       ; 20/08/2017
00015A3B 803D[C89C0100]10   <1>       cmp    byte [audio_bps], 16
00015A42 750F               <1>       jne    short sb16_gcd_1 ; 8 bit DMA channel
00015A44 E4C6               <1>       in     al, 0C6h ; DMA channel 5 count register
00015A46 88C2               <1>       mov    dl, al
00015A48 E4C6               <1>       in     al, 0C6h
00015A4A 88C6               <1>       mov     dh, al
00015A4C 0FB7C2             <1>       movzx  eax, dx
00015A4F D1E0               <1>       shl    eax, 1 ; word count -> byte count
00015A51 EB4E               <1>       jmp    short sb16_gcd_2
                            <1> sb16_gcd_1:
00015A53 E403               <1>       in     al, 03h ; DMA channel 1 count register
00015A55 88C2               <1>       mov    dl, al
00015A57 E403               <1>       in     al, 03h
00015A59 88C6               <1>       mov     dh, al
00015A5B 0FB7C2             <1>       movzx  eax, dx
00015A5E EB41               <1>       jmp    short sb16_gcd_2
                            <1> ;sb16_gcd_2:
                            <1> ;     cmp    eax, ecx
                            <1> ;     jnb    short sb16_gcd_3
                            <1> ;     ; remain count < graphics bytes
                            <1> ;     mov    eax, ecx ; fix remain count to data size
                            <1> ;sb16_gcd_3:
                            <1> ;     sub    edi, eax
                            <1> ;     jna    short sb16_gcd_4
                            <1> ;     add    esi, edi ; dma buffer offset
                            <1> ;sb16_gcd_4:
```

```
2745                            <1> ;       mov     edi, ebx ; buffer address (for graphics)
2746                            <1> ;       mov     [u.r0], ecx
2747                            <1> ;       rep     movsb
2748                            <1> ;       retn
2749                            <1>
2750                            <1> get_current_sound_data:
2751                            <1>         ; 24/06/2017
2752                            <1>         ; 22/06/2017
2753                            <1>         ; get current sound (PCM out) data for graphics
2754                            <1>         ;
2755                            <1>         ; ebx = Physical address (on page boundary)
2756                            <1>         ; ecx = Byte count
2757                            <1>         ; [audio_buff_size]
2758                            <1>
2759                            <1>         ;mov   edi, [audio_buff_size]
2760 00015A60 8B3D[BC9C0100]      <1>         mov     edi, [audio_dmabuff_size]
2761 00015A66 8B35[B89C0100]      <1>         mov     esi, [audio_dma_buff]
2762 00015A6C 803D[999C0100]02    <1>         cmp     byte [audio_device], 2
2763 00015A73 72C0                <1>         jb      short sb16_current_sound_data ; = 1
2764 00015A75 D1EF                <1>         shr     edi, 1
2765 00015A77 39CF                <1>         cmp     edi, ecx
2766 00015A79 7302                <1>         jnb     short gcd_0
2767 00015A7B 89F9                <1>         mov     ecx, edi
2768                            <1> gcd_0:
2769 00015A7D 803D[999C0100]03    <1>         cmp     byte [audio_device], 3
2770 00015A84 7232                <1>         jb      short ac97_current_sound_data ; = 2
2771                            <1>         ; = 3
2772                            <1> vt8233_current_sound_data:
2773                            <1>         ; 22/06/2017
2774                            <1>         ; 21/06/2017
2775                            <1>         ; get current sound (PCM out) data for graphics
2776                            <1>         ; (for VT 8233, VT 8237R)
2777                            <1>         ; ebx = Physical address (on page boundary)
2778                            <1>         ; ecx = Byte count
2779                            <1>         ; [audio_buff_size]
2780                            <1>
2781                            <1>         ;;mov  edi, [audio_buff_size]
2782                            <1>         ;mov   edi, [audio_dmabuff_size]
2783                            <1>         ;mov   esi, [audio_dma_buff]
2784                            <1>         ;shr   edi, 1
2785                            <1>         ;cmp   edi, ecx
2786                            <1>         ;jnb   short vt8233_gcd_1
2787                            <1>         ;mov   ecx, edi
2788                            <1> vt8233_gcd_1:
2789 00015A86 BA0C000000          <1>         mov     edx, VIA_REG_OFFSET_CURR_COUNT
2790 00015A8B E88FF5FFFF          <1>         call    ctrl_io_r32
2791 00015A90 89C2                <1>         mov     edx, eax ; remain count (bits 23-0),
2792                            <1>                          ; SGD index (bits 31-24)
2793 00015A92 81E200000001        <1>         and     edx, 1000000h ; SGD index (0 = 1st half)
2794 00015A98 7402                <1>         jz      short vt8233_gcd_2
2795                            <1>         ; the second half of DMA buffer
2796 00015A9A 01FE                <1>         add     esi, edi
2797                            <1> vt8233_gcd_2:
2798 00015A9C 25FFFFFF00          <1>         and     eax, 0FFFFFFh ; bits 23-0
2799                            <1> ac97_gcd_2:
2800                            <1> sb16_gcd_2:
2801 00015AA1 39C8                <1>         cmp     eax, ecx
2802 00015AA3 7302                <1>         jnb     short vt8233_gcd_3
2803                            <1>         ; remain count < graphics bytes
2804 00015AA5 89C8                <1>         mov     eax, ecx ; fix remain count to data size
2805                            <1> vt8233_gcd_3:
2806 00015AA7 29C7                <1>         sub     edi, eax
2807 00015AA9 7602                <1>         jna     short vt8233_gcd_4
2808 00015AAB 01FE                <1>         add     esi, edi ; dma buffer offset
2809                            <1> vt8233_gcd_4:
2810 00015AAD 89DF                <1>         mov     edi, ebx ; buffer address (for graphics)
2811 00015AAF 890D[64030300]      <1>         mov     [u.r0], ecx
2812 00015AB5 F3A4                <1>         rep     movsb
2813                            <1> vt8233_gcd_5:
2814 00015AB7 C3                  <1>         retn
2815                            <1>
2816                            <1> ac97_current_sound_data:
2817                            <1>         ; 23/06/2017
2818                            <1>         ; 22/06/2017
2819                            <1>         ; get current sound (PCM out) data for graphics
2820                            <1>         ; (for AC'97, ICH)
2821                            <1>         ; ebx = Physical address (on page boundary)
2822                            <1>         ; ecx = Byte count
2823                            <1>         ; [audio_buff_size]
2824                            <1>
2825                            <1>         ;;mov  edi, [audio_buff_size]
2826                            <1>         ;mov   edi, [audio_dmabuff_size]
2827                            <1>         ;mov   esi, [audio_dma_buff]
2828                            <1>         ;shr   edi, 1
2829                            <1>         ;cmp   edi, ecx
2830                            <1>         ;jnb   short ac97_gcd_0
2831                            <1>         ;mov   ecx, edi
2832                            <1> ac97_gcd_0:
2833 00015AB8 66BA1400            <1>         mov     dx, PO_CIV_REG ; Position In Current Buff Reg
2834 00015ABC 660315[D29C0100]    <1>         add     dx, [NABMBAR]
2835 00015AC3 EC                  <1>         in      al, dx ; current index value
2836 00015AC4 A801                <1>         test    al, 1
2837 00015AC6 7402                <1>         jz      short ac97_gcd_1
2838 00015AC8 01FE                <1>         add     esi, edi
2839                            <1> ac97_gcd_1:
2840 00015ACA 31C0                <1>         xor     eax, eax
2841 00015ACC 66BA1800            <1>         mov     dx, PO_PICB_REG ; Position In Current Buff Reg
2842 00015AD0 660315[D29C0100]    <1>         add     dx, [NABMBAR]
2843 00015AD7 66ED                <1>         in      ax, dx ; remain dwords
2844 00015AD9 C1E002              <1>         shl     eax, 2 ; remain bytes ; 23/06/2017
2845 00015ADC EBC3                <1>         jmp     short ac97_gcd_2
2846                            <1> ;       cmp     eax, ecx
2847                            <1> ;       jnb     short ac97_gcd_2
2848                            <1> ;       ; remain count < graphics bytes
2849                            <1> ;       mov     eax, ecx ; fix remain count to data size
```

```
                            <1> ;ac97_gcd_2:
                            <1> ;       sub     edi, eax
                            <1> ;       jna     short ac97_gcd_3
                            <1> ;       add     esi, edi ; dma buffer offset
                            <1> ;ac97_gcd_3:
                            <1> ;       mov     edi, ebx ; buffer address (for graphics)
                            <1> ;       mov     [u.r0], ecx
                            <1> ;       rep     movsb
                            <1> ;       retn
                            <1>
                            <1> sb16_get_dma_buff_off:
                            <1>         ; 28/10/2017
                            <1>         ; 24/06/2017
                            <1>         ; 22/06/2017
                            <1>         ; get current (PCM OUT DMA buffer) pointer
                            <1>         ; (for Sound Blaster 16)
                            <1>
                            <1>         ;mov   ecx, [audio_dmabuff_size]
                            <1>         ;xor   ebx, ebx
                            <1>         ;shr   ecx, 1
                            <1> sb16_gdmabo_0:
                            <1>         ; 28/10/2017
00015ADE 803D[C89C0100]10   <1>         cmp     byte [audio_bps], 16
00015AE5 750F               <1>         jne     short sb16_gdmabo_1 ; 8 bit DMA channel
                            <1>         ; 16 bit DMA channel
00015AE7 E4C6               <1>         in      al, 0C6h ; DMA channel 5 count register
00015AE9 88C2               <1>         mov     dl, al
00015AEB E4C6               <1>         in      al, 0C6h
00015AED 88C6               <1>         mov     dh, al
00015AEF 0FB7C2             <1>         movzx   eax, dx
00015AF2 D1E0               <1>         shl     eax, 1 ; word count -> byte count
00015AF4 EB3D               <1>         jmp     short sb16_gdmabo_2
                            <1> sb16_gdmabo_1:
00015AF6 E403               <1>         in      al, 03h ; DMA channel 1 count register
00015AF8 88C2               <1>         mov     dl, al
00015AFA E403               <1>         in      al, 03h
00015AFC 88C6               <1>         mov     dh, al
00015AFE 0FB7C2             <1>         movzx   eax, dx
00015B01 EB30               <1>         jmp     short sb16_gdmabo_2
                            <1>
                            <1> get_dma_buffer_offset:
                            <1>         ; 24/06/2017
                            <1>         ; 22/06/2017
                            <1>         ; get current sound (PCM out) data for graphics
                            <1>         ;
                            <1>         ; ebx = Physical address (on page boundary)
                            <1>         ; ecx = Byte count
                            <1>         ; [audio_buff_size]
                            <1>
00015B03 8B0D[BC9C0100]     <1>         mov     ecx, [audio_dmabuff_size]
00015B09 31DB               <1>         xor     ebx, ebx
                            <1> gdmabo_0:
00015B0B 803D[999C0100]02   <1>         cmp     byte [audio_device], 2
00015B12 72CA               <1>         jb      short sb16_get_dma_buff_off
00015B14 742A               <1>         je      short ac97_get_dma_buff_off
                            <1>
                            <1> vt8233_get_dma_buff_off:
                            <1>         ; 24/06/2017
                            <1>         ; 22/06/2017
                            <1>         ; get current (PCM OUT DMA buffer) pointer
                            <1>         ; (for VT 8233, VT 8237R)
                            <1>
                            <1>         ;mov   ecx, [audio_dmabuff_size]
                            <1>         ;xor   ebx, ebx
00015B16 D1E9               <1>         shr     ecx, 1
                            <1> vt8233_gdmabo_0:
00015B18 BA0C000000         <1>         mov     edx, VIA_REG_OFFSET_CURR_COUNT
00015B1D E8FDF4FFFF         <1>         call    ctrl_io_r32
00015B22 89C2               <1>         mov     edx, eax ; remain count (bits 23-0),
                            <1>                          ; SGD index (bits 31-24)
00015B24 81E200000001       <1>         and     edx, 1000000h ; SGD index (0 = 1st half)
00015B2A 7402               <1>         jz      short vt8233_gdmabo_1
                            <1>         ; the second half of DMA buffer
00015B2C 89CB               <1>         mov     ebx, ecx
                            <1> vt8233_gdmabo_1:
00015B2E 25FFFFFF00         <1>         and     eax, 0FFFFFFh ; bits 23-0
                            <1> sb16_gdmabo_2:
                            <1> ac97_gdmabo_2:
00015B33 29C1               <1>         sub     ecx, eax
00015B35 7602               <1>         jna     short vt8233_gdmabo_2
00015B37 01CB               <1>         add     ebx, ecx ; dma buffer offset
                            <1> vt8233_gdmabo_2:
00015B39 891D[64030300]     <1>         mov     [u.r0], ebx
00015B3F C3                 <1>         retn
                            <1>
                            <1> ac97_get_dma_buff_off:
                            <1>         ; 24/06/2017
                            <1>         ; 22/06/2017
                            <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
00015B40 D1E9               <1>         shr     ecx, 1
                            <1> ac97_gdmabo_0:
00015B42 66BA1400           <1>         mov     dx, PO_CIV_REG ; Position In Current Buff Reg
00015B46 660315[D29C0100]   <1>         add     dx, [NABMBAR]
00015B4D EC                 <1>         in      al, dx ; current index value
00015B4E A801               <1>         test    al, 1
00015B50 7402               <1>         jz      short ac97_gdmabo_1
00015B52 89CB               <1>         mov     ebx, ecx
                            <1> ac97_gdmabo_1:
```

```
2955 00015B54 31C0                <1>       xor    eax, eax
2956 00015B56 66BA1800            <1>       mov    dx, PO_PICB_REG ; Position In Current Buff Reg
2957 00015B5A 660315[D29C0100]    <1>       add    dx, [NABMBAR]
2958 00015B61 66ED                <1>       in     ax, dx ; remain dwords
2959 00015B63 EBCE                <1>       jmp    short ac97_gdmabo_2
3433
3434 00015B65 90<rept>                 align 4
3435
3436                                   %include 'vgadata.s' ; 04/07/2016
   1                              <1> ; ****************************************************************************************************
   2                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.3 - vgadata.s  (palette and font data)
   3                              <1> ; ----------------------------------------------------------------------------
   4                              <1> ; Last Update: 01/01/2021
   5                              <1> ; ----------------------------------------------------------------------------
   6                              <1> ; Beginning: 16/01/2016
   7                              <1> ; ----------------------------------------------------------------------------
   8                              <1> ; Assembler: NASM version 2.15 (trdos386.s)
   9                              <1> ; ----------------------------------------------------------------------------
  10                              <1> ; Turkish Rational DOS
  11                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
  12                              <1> ;
  13                              <1> ; Derived from 'Plex86/Bochs VGABios' source code, vgabios-0.7a (2011)
  14                              <1> ; by the LGPL VGABios Developers Team (2001-2008), 'vgatables.h'
  15                              <1> ;
  16                              <1> ; Oracle VirtualBox 5.0.24 VGABios Source Code
  17                              <1> ; ('vgabios.c', 'vgatables.h', 'vgafonts.h', 'vgarom.asm')
  18                              <1> ;
  19                              <1> ; Palette and font data in assembly language format:
  20                              <1> ; 'VBoxVgaBiosAlternative.asm'
  21                              <1>
  22                              <1> ; ****************************************************************************************************
  23                              <1>
  24                              <1> ; 25/11/2020 (TRDOS 386 v2.0.3)
  25                              <1> ; ('vgatables.h' - 30/12/2019 - vruppert)
  26                              <1>
  27                              <1> ; 04/07/2016
  28                              <1> ; COLOR DATA
  29                              <1>
  30                              <1> palette0: ; (63+1)*3
31 00015B68 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
31 00015B71 00000000000000      <1>
32 00015B78 00000000000000002A- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
32 00015B81 2A2A2A2A2A2A2A      <1>
33 00015B88 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
33 00015B91 2A2A2A2A2A2A2A      <1>
34 00015B98 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
34 00015BA1 2A2A2A2A2A2A2A      <1>
35 00015BA8 2A2A2A2A2A2A2A2A3F- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
35 00015BB1 3F3F3F3F3F3F3F      <1>
36 00015BB8 3F3F3F3F3F3F3F3F3F- <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
36 00015BC1 3F3F3F3F3F3F3F      <1>
37 00015BC8 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
37 00015BD1 00000000000000      <1>
38 00015BD8 00000000000000002A- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
38 00015BE1 2A2A2A2A2A2A2A      <1>
39 00015BE8 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
39 00015BF1 2A2A2A2A2A2A2A      <1>
40 00015BF8 2A2A2A2A2A2A2A2A2A- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah
40 00015C01 2A2A2A2A2A2A2A      <1>
41 00015C08 2A2A2A2A2A2A2A2A3F- <1>     db  02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 02ah, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
41 00015C11 3F3F3F3F3F3F3F      <1>
42 00015C18 3F3F3F3F3F3F3F3F3F- <1>     db  03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh, 03fh
42 00015C21 3F3F3F3F3F3F3F      <1>
43                              <1> palette1: ; (63+1)*3
44 00015C28 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
44 00015C31 002A2A2A00002A      <1>
45 00015C38 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
45 00015C41 000000002A002A      <1>
46 00015C48 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
46 00015C51 2A2A15002A2A2A      <1>
47 00015C58 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
47 00015C61 153F3F3F15153F      <1>
48 00015C68 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
48 00015C71 151515153F153F      <1>
49 00015C78 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
49 00015C81 3F3F3F153F3F3F      <1>
50 00015C88 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
50 00015C91 002A2A2A00002A      <1>
51 00015C98 002A2A15002A2A2A00- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah
51 00015CA1 000000002A002A      <1>
52 00015CA8 00002A2A2A00002A00- <1>     db  000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah, 000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah
52 00015CB1 2A2A15002A2A2A      <1>
53 00015CB8 15151515153F153F15- <1>     db  015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh, 015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh
53 00015CC1 153F3F3F15153F      <1>
54 00015CC8 153F3F3F153F3F3F15- <1>     db  015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
54 00015CD1 151515153F153F      <1>
55 00015CD8 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
55 00015CE1 3F3F3F153F3F3F      <1>
56                              <1> palette2: ; (63+1)*3
57 00015CE8 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
57 00015CF1 002A2A2A00002A      <1>
58 00015CF8 002A2A2A002A2A2A00- <1>     db  000h, 02ah, 02ah, 02ah, 000h, 02ah, 02ah, 02ah, 000h, 000h, 015h, 000h, 000h, 03fh, 000h, 02ah
58 00015D01 001500003F002A      <1>
59 00015D08 15002A3F2A00152A00- <1>     db  015h, 000h, 02ah, 03fh, 02ah, 000h, 015h, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 02ah, 02ah, 03fh
59 00015D11 3F2A2A152A2A3F      <1>
60 00015D18 00150000152A003F00- <1>     db  000h, 015h, 000h, 000h, 015h, 02ah, 000h, 03fh, 000h, 000h, 03fh, 02ah, 02ah, 015h, 000h, 02ah
60 00015D21 003F2A2A15002A      <1>
61 00015D28 152A2A3F002A3F2A00- <1>     db  015h, 02ah, 02ah, 03fh, 000h, 02ah, 03fh, 02ah, 000h, 015h, 015h, 000h, 015h, 03fh, 000h, 03fh
61 00015D31 151500153F003F      <1>
62 00015D38 15003F3F2A15152A15- <1>     db  015h, 000h, 03fh, 03fh, 02ah, 015h, 015h, 02ah, 015h, 03fh, 02ah, 03fh, 015h, 02ah, 03fh, 03fh
62 00015D41 3F2A3F152A3F3F      <1>
63 00015D48 15000015002A152A00- <1>     db  015h, 000h, 000h, 015h, 000h, 02ah, 015h, 02ah, 000h, 015h, 02ah, 02ah, 03fh, 000h, 000h, 03fh
63 00015D51 152A2A3F00003F      <1>
64 00015D58 002A3F2A003F2A2A15- <1>     db  000h, 02ah, 03fh, 02ah, 000h, 03fh, 02ah, 02ah, 015h, 000h, 015h, 015h, 000h, 03fh, 015h, 02ah
64 00015D61 001515003F152A      <1>
65 00015D68 15152A3F3F00153F00- <1>     db  015h, 015h, 02ah, 03fh, 03fh, 000h, 015h, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 03fh, 02ah, 03fh
65 00015D71 3F3F2A153F2A3F      <1>
66 00015D78 15150015152A153F00- <1>     db  015h, 015h, 000h, 015h, 015h, 02ah, 015h, 03fh, 000h, 015h, 03fh, 02ah, 03fh, 015h, 000h, 03fh
66 00015D81 153F2A3F15003F      <1>
67 00015D88 152A3F3F003F3F2A15- <1>     db  015h, 02ah, 03fh, 03fh, 000h, 03fh, 03fh, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
```

```
67 00015D91 151515153F153F      <1>
68 00015D98 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
68 00015DA1 3F3F3F153F3F3F      <1>
69                              <1> palette3: ; 256*3
70 00015DA8 00000000002A002A00- <1>     db  000h, 000h, 000h, 000h, 000h, 02ah, 000h, 02ah, 000h, 000h, 02ah, 02ah, 02ah, 000h, 000h, 02ah
70 00015DB1 002A2A2A00002A      <1>
71 00015DB8 002A2A15002A2A2A15- <1>     db  000h, 02ah, 02ah, 015h, 000h, 02ah, 02ah, 02ah, 015h, 015h, 015h, 015h, 015h, 03fh, 015h, 03fh
71 00015DC1 151515153F153F      <1>
72 00015DC8 15153F3F3F15153F15- <1>     db  015h, 015h, 03fh, 03fh, 03fh, 015h, 015h, 03fh, 015h, 03fh, 03fh, 03fh, 015h, 03fh, 03fh, 03fh
72 00015DD1 3F3F3F153F3F3F      <1>
73 00015DD8 000000050505080808- <1>     db  000h, 000h, 000h, 005h, 005h, 005h, 008h, 008h, 008h, 00bh, 00bh, 00bh, 00eh, 00eh, 00eh, 011h
73 00015DE1 0B0B0B0E0E0E11      <1>
74 00015DE8 11111414141818181C- <1>     db  011h, 011h, 014h, 014h, 014h, 018h, 018h, 018h, 01ch, 01ch, 01ch, 020h, 020h, 020h, 024h, 024h
74 00015DF1 1C1C2020202424      <1>
75 00015DF8 242828282D2D2D3232- <1>     db  024h, 028h, 028h, 028h, 02dh, 02dh, 02dh, 032h, 032h, 032h, 038h, 038h, 038h, 03fh, 03fh, 03fh
75 00015E01 323838383F3F3F      <1>
76 00015E08 00003F10003F1F003F- <1>     db  000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 03fh, 03fh
76 00015E11 2F003F3F003F3F      <1>
77 00015E18 002F3F001F3F00103F- <1>     db  000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh
77 00015E21 00003F10003F1F      <1>
78 00015E28 003F2F003F3F002F3F- <1>     db  000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h, 02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 000h
78 00015E31 001F3F00103F00      <1>
79 00015E38 003F00003F10003F1F- <1>     db  000h, 03fh, 000h, 000h, 03fh, 010h, 000h, 03fh, 01fh, 000h, 03fh, 02fh, 000h, 03fh, 03fh, 000h
79 00015E41 003F2F003F3F00      <1>
80 00015E48 2F3F001F3F00103F1F- <1>     db  02fh, 03fh, 000h, 01fh, 03fh, 000h, 010h, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh
80 00015E51 1F3F271F3F2F1F      <1>
81 00015E58 3F371F3F3F1F3F3F1F- <1>     db  03fh, 037h, 01fh, 03fh, 03fh, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h
81 00015E61 373F1F2F3F1F27      <1>
82 00015E68 3F1F1F3F271F3F2F1F- <1>     db  03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh, 02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h
82 00015E71 3F371F3F3F1F37      <1>
83 00015E78 3F1F2F3F1F273F1F1F- <1>     db  03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh, 01fh, 01fh, 03fh, 01fh, 01fh, 03fh, 027h, 01fh, 03fh
83 00015E81 3F1F1F3F271F3F      <1>
84 00015E88 2F1F3F371F3F3F1F37- <1>     db  02fh, 01fh, 03fh, 037h, 01fh, 03fh, 03fh, 01fh, 037h, 03fh, 01fh, 02fh, 03fh, 01fh, 027h, 03fh
84 00015E91 3F1F2F3F1F273F      <1>
85 00015E98 2D2D3F312D3F362D3F- <1>     db  02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03fh, 03fh
85 00015EA1 3A2D3F3F2D3F3F      <1>
86 00015EA8 2D3A3F2D363F2D313F- <1>     db  02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h
86 00015EB1 2D2D3F312D3F36      <1>
87 00015EB8 2D3F3A2D3F3F2D3A3F- <1>     db  02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh, 03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 02dh
87 00015EC1 2D363F2D313F2D      <1>
88 00015EC8 2D3F2D2D3F312D3F36- <1>     db  02dh, 03fh, 02dh, 02dh, 03fh, 031h, 02dh, 03fh, 036h, 02dh, 03fh, 03ah, 02dh, 03fh, 03fh, 02dh
88 00015ED1 2D3F3A2D3F3F2D      <1>
89 00015ED8 3A3F2D363F2D313F00- <1>     db  03ah, 03fh, 02dh, 036h, 03fh, 02dh, 031h, 03fh, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h
89 00015EE1 001C07001C0E00      <1>
90 00015EE8 1C15001C1C001C1C00- <1>     db  01ch, 015h, 000h, 01ch, 01ch, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h
90 00015EF1 151C000E1C0007      <1>
91 00015EF8 1C00001C07001C0E00- <1>     db  01ch, 000h, 000h, 01ch, 007h, 000h, 01ch, 00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h
91 00015F01 1C15001C1C0015      <1>
92 00015F08 1C000E1C00071C0000- <1>     db  01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch, 000h, 000h, 01ch, 000h, 000h, 01ch, 007h, 000h, 01ch
92 00015F11 1C00001C07001C      <1>
93 00015F18 0E001C15001C1C0015- <1>     db  00eh, 000h, 01ch, 015h, 000h, 01ch, 01ch, 000h, 015h, 01ch, 000h, 00eh, 01ch, 000h, 007h, 01ch
93 00015F21 1C000E1C00071C      <1>
94 00015F28 0E0E1C110E1C150E1C- <1>     db  00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 01ch, 01ch
94 00015F31 180E1C1C0E1C1C      <1>
95 00015F38 0E181C0E151C0E111C- <1>     db  00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h
95 00015F41 0E0E1C110E1C15      <1>
96 00015F48 0E1C180E1C1C0E181C- <1>     db  00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh, 018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 00eh
96 00015F51 0E151C0E111C0E      <1>
97 00015F58 0E1C0E0E1C110E1C15- <1>     db  00eh, 01ch, 00eh, 00eh, 01ch, 011h, 00eh, 01ch, 015h, 00eh, 01ch, 018h, 00eh, 01ch, 01ch, 00eh
97 00015F61 0E1C180E1C1C0E      <1>
98 00015F68 181C0E151C0E111C14- <1>     db  018h, 01ch, 00eh, 015h, 01ch, 00eh, 011h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h
98 00015F71 141C16141C1814      <1>
99 00015F78 1C1A141C1C141C1C14- <1>     db  01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h
99 00015F81 1A1C14181C1416      <1>
100 00015F88 1C14141C16141C1814- <1>    db  01ch, 014h, 014h, 01ch, 016h, 014h, 01ch, 018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah
100 00015F91 1C1A141C1C141A      <1>
101 00015F98 1C14181C14161C1414- <1>    db  01ch, 014h, 018h, 01ch, 014h, 016h, 01ch, 014h, 014h, 01ch, 014h, 014h, 01ch, 016h, 014h, 01ch
101 00015FA1 1C14141C16141C      <1>
102 00015FA8 18141C1A141C1C141A- <1>    db  018h, 014h, 01ch, 01ah, 014h, 01ch, 01ch, 014h, 01ah, 01ch, 014h, 018h, 01ch, 014h, 016h, 01ch
102 00015FB1 1C14181C14161C      <1>
103 00015FB8 000010040010080010- <1>    db  000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h, 010h, 010h
103 00015FC1 0C001010001010      <1>
104 00015FC8 000C10000810000410- <1>    db  000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h
104 00015FD1 00001004001008      <1>
105 00015FD8 00100C001010000C10- <1>    db  000h, 010h, 00ch, 000h, 010h, 010h, 000h, 00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 000h
105 00015FE1 00081000041000      <1>
106 00015FE8 0010000010040010008- <1>    db  000h, 010h, 000h, 000h, 010h, 004h, 000h, 010h, 008h, 000h, 010h, 00ch, 000h, 010h, 010h, 000h
106 00015FF1 00100C00101000      <1>
107 00015FF8 0C10000810000410008- <1>    db  00ch, 010h, 000h, 008h, 010h, 000h, 004h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h
107 00016001 08100A08100C08      <1>
108 00016008 100E081010081010008- <1>    db  010h, 00eh, 008h, 010h, 010h, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah
108 00016011 0E10080C10080A      <1>
109 00016018 100808100A08100C08- <1>    db  010h, 008h, 008h, 010h, 00ah, 008h, 010h, 00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh
109 00016021 100E081010080E      <1>
110 00016028 10080C10080A100808- <1>    db  010h, 008h, 00ch, 010h, 008h, 00ah, 010h, 008h, 008h, 010h, 008h, 008h, 010h, 00ah, 008h, 010h
110 00016031 100808100A0810      <1>
111 00016038 0C08100E081010080E- <1>    db  00ch, 008h, 010h, 00eh, 008h, 010h, 010h, 008h, 00eh, 010h, 008h, 00ch, 010h, 008h, 00ah, 010h
111 00016041 10080C10080A10      <1>
112 00016048 0B0B100C0B100D0B10- <1>    db  00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 010h, 010h
112 00016051 0F0B10100B1010      <1>
113 00016058 0B0F100B0D100B0C10- <1>    db  00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh
113 00016061 0B0B100C0B100D      <1>
114 00016068 0B100F0B10100B0F10- <1>    db  00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh, 00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 00bh
114 00016071 0B0D100B0C100B      <1>
115 00016078 0B100B0B100C0B100D- <1>    db  00bh, 010h, 00bh, 00bh, 010h, 00ch, 00bh, 010h, 00dh, 00bh, 010h, 00fh, 00bh, 010h, 010h, 00bh
115 00016081 0B100F0B10100B      <1>
116 00016088 0F100B0D100B0C1000- <1>    db  00fh, 010h, 00bh, 00dh, 010h, 00bh, 00ch, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
116 00016091 00000000000000      <1>
117 00016098 0000000000000000000- <1>   db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
117 000160A1 00000000000000      <1>
118                              <1>
119                              <1>
120                              <1> ; 04/07/2016
121                              <1> ; FONT DATA
122                              <1>
123                              <1> CRT_CHAR_GEN:
124                              <1> vgafont8:
125 000160A8 00000000000000007E- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 081h, 0a5h, 081h, 0bdh, 099h, 081h, 07eh
125 000160B1 81A581BD99817E      <1>
126 000160B8 7EFFDBFFC3E7FF7E6C- <1>    db  07eh, 0ffh, 0dbh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 06ch, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h
126 000160C1 FEFEFE7C381000      <1>
```

```
127 000160C8 10387CFE7C38100038- <1>     db  010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 038h, 07ch, 038h, 0feh, 0feh, 07ch, 038h, 07ch
127 000160D1 7C38FEFE7C387C      <1>
128 000160D8 1010387CFE7C387C00- <1>     db  010h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 07ch, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h
128 000160E1 00183C3C180000      <1>
129 000160E8 FFFFE7C3C3E7FFFF00- <1>     db  0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h
129 000160F1 3C664242663C00      <1>
130 000160F8 FFC399BDBD99C3FF0F- <1>     db  0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 00fh, 007h, 00fh, 07dh, 0cch, 0cch, 0cch, 078h
130 00016101 070F7DCCCCCC78      <1>
131 00016108 3C6666663C187E183F- <1>     db  03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 03fh, 033h, 03fh, 030h, 030h, 070h, 0f0h, 0e0h
131 00016111 333F303070F0E0      <1>
132 00016118 7F637F636367E6C099- <1>     db  07fh, 063h, 07fh, 063h, 063h, 067h, 0e6h, 0c0h, 099h, 05ah, 03ch, 0e7h, 0e7h, 03ch, 05ah, 099h
132 00016121 5A3CE7E73C5A99      <1>
133 00016128 80E0F8FEF8E0800002- <1>     db  080h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 080h, 000h, 002h, 00eh, 03eh, 0feh, 03eh, 00eh, 002h, 000h
133 00016131 0E3EFE3E0E0200      <1>
134 00016138 183C7E18187E3C1866- <1>     db  018h, 03ch, 07eh, 018h, 018h, 07eh, 03ch, 018h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 000h
134 00016141 66666666006600      <1>
135 00016148 7FDBDB7B1B1B1B003E- <1>     db  07fh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 000h, 03eh, 063h, 038h, 06ch, 06ch, 038h, 0cch, 078h
135 00016151 63386C6C38CC78      <1>
136 00016158 000000007E7E7E0018- <1>     db  000h, 000h, 000h, 000h, 07eh, 07eh, 07eh, 000h, 018h, 03ch, 07eh, 018h, 07eh, 03ch, 018h, 0ffh
136 00016161 3C7E187E3C18FF      <1>
137 00016168 183C7E1818181800018- <1>     db  018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h
137 00016171 1818187E3C1800      <1>
138 00016178 00180CFE0C18000000- <1>     db  000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h
138 00016181 3060FE60300000      <1>
139 00016188 0000C0C0C0FE000000- <1>     db  000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h
139 00016191 2466FF66240000      <1>
140 00016198 00183C7EFFFF000000- <1>     db  000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 000h, 000h, 000h, 0ffh, 0ffh, 07eh, 03ch, 018h, 000h, 000h
140 000161A1 FFFF7E3C180000      <1>
141 000161A8 0000000000000000030- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 078h, 078h, 030h, 030h, 000h, 030h, 000h
141 000161B1 78783030003000      <1>
142 000161B8 6C6C6C000000000006C- <1>     db  06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 0feh, 06ch, 06ch, 000h
142 000161C1 6CFE6CFE6C6C00      <1>
143 000161C8 307CC0780CF8300000- <1>     db  030h, 07ch, 0c0h, 078h, 00ch, 0f8h, 030h, 000h, 000h, 0c6h, 0cch, 018h, 030h, 066h, 0c6h, 000h
143 000161D1 C6CC183066C600      <1>
144 000161D8 386C3876DCCC760060- <1>     db  038h, 06ch, 038h, 076h, 0dch, 0cch, 076h, 000h, 060h, 060h, 0c0h, 000h, 000h, 000h, 000h, 000h
144 000161E1 60C00000000000      <1>
145 000161E8 183060606030180060- <1>     db  018h, 030h, 060h, 060h, 060h, 030h, 018h, 000h, 060h, 030h, 018h, 018h, 018h, 030h, 060h, 000h
145 000161F1 30181818306000      <1>
146 000161F8 00663CFF3C66000000- <1>     db  000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 000h
146 00016201 3030FC30300000      <1>
147 00016208 0000000000030306000- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 060h, 000h, 000h, 000h, 0fch, 000h, 000h, 000h, 000h
147 00016211 0000FC00000000      <1>
148 00016218 0000000000030300006- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 030h, 000h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h
148 00016221 0C183060C08000      <1>
149 00016228 7CC6CEDEF6E67C0030- <1>     db  07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 07ch, 000h, 030h, 070h, 030h, 030h, 030h, 030h, 0fch, 000h
149 00016231 7030303030FC00      <1>
150 00016238 78CC0C3860CCFC0078- <1>     db  078h, 0cch, 00ch, 038h, 060h, 0cch, 0fch, 000h, 078h, 0cch, 00ch, 038h, 00ch, 0cch, 078h, 000h
150 00016241 CC0C380CCC7800      <1>
151 00016248 1C3C6CCCFE0C1E00FC- <1>     db  01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 01eh, 000h, 0fch, 0c0h, 0f8h, 00ch, 00ch, 0cch, 078h, 000h
151 00016251 C0F80C0CCC7800      <1>
152 00016258 3860C0F8CCCC7800FC- <1>     db  038h, 060h, 0c0h, 0f8h, 0cch, 0cch, 078h, 000h, 0fch, 0cch, 00ch, 018h, 030h, 030h, 030h, 000h
152 00016261 CC0C1830303000      <1>
153 00016268 78CCCC78CCCC780078- <1>     db  078h, 0cch, 0cch, 078h, 0cch, 0cch, 078h, 000h, 078h, 0cch, 0cch, 07ch, 00ch, 018h, 070h, 000h
153 00016271 CCCC7C0C187000      <1>
154 00016278 0030300000030300000- <1>     db  000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 000h, 000h, 030h, 030h, 060h
154 00016281 30300000303060      <1>
155 00016288 183060C06030180000- <1>     db  018h, 030h, 060h, 0c0h, 060h, 030h, 018h, 000h, 000h, 000h, 0fch, 000h, 000h, 0fch, 000h, 000h
155 00016291 00FC0000FC0000      <1>
156 00016298 6030180C1830600078- <1>     db  060h, 030h, 018h, 00ch, 018h, 030h, 060h, 000h, 078h, 0cch, 00ch, 018h, 030h, 000h, 030h, 000h
156 000162A1 CC0C1830003000      <1>
157 000162A8 7CC6DEDEDEC0780030- <1>     db  07ch, 0c6h, 0deh, 0deh, 0deh, 0c0h, 078h, 000h, 030h, 078h, 0cch, 0cch, 0fch, 0cch, 0cch, 000h
157 000162B1 78CCCCFCCCCC00      <1>
158 000162B8 FC66667C6666FC003C- <1>     db  0fch, 066h, 066h, 07ch, 066h, 066h, 0fch, 000h, 03ch, 066h, 0c0h, 0c0h, 0c0h, 066h, 03ch, 000h
158 000162C1 66C0C0C0663C00      <1>
159 000162C8 F86C6666666CF800FE- <1>     db  0f8h, 06ch, 066h, 066h, 066h, 06ch, 0f8h, 000h, 0feh, 062h, 068h, 078h, 068h, 062h, 0feh, 000h
159 000162D1 6268786862FE00      <1>
160 000162D8 FE6268786860F0003C- <1>     db  0feh, 062h, 068h, 078h, 068h, 060h, 0f0h, 000h, 03ch, 066h, 0c0h, 0c0h, 0ceh, 066h, 03eh, 000h
160 000162E1 66C0C0CE663E00      <1>
161 000162E8 CCCCCCFCCCCCCC0078- <1>     db  0cch, 0cch, 0cch, 0fch, 0cch, 0cch, 0cch, 000h, 078h, 030h, 030h, 030h, 030h, 030h, 078h, 000h
161 000162F1 30303030307800      <1>
162 000162F8 1E0C0C0CCCCC7800E6- <1>     db  01eh, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 0e6h, 066h, 06ch, 078h, 06ch, 066h, 0e6h, 000h
162 00016301 666C786C66E600      <1>
163 00016308 F06060606266FE00C6- <1>     db  0f0h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 000h
163 00016311 EEFEFED6C6C600      <1>
164 00016318 C6E6F6DECEC6C60038- <1>     db  0c6h, 0e6h, 0f6h, 0deh, 0ceh, 0c6h, 0c6h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h
164 00016321 6CC6C6C66C3800      <1>
165 00016328 FC66667C6060F00078- <1>     db  0fch, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 078h, 0cch, 0cch, 0cch, 0dch, 078h, 01ch, 000h
165 00016331 CCCCCCDC781C00      <1>
166 00016338 FC66667C6C66E60078- <1>     db  0fch, 066h, 066h, 07ch, 06ch, 066h, 0e6h, 000h, 078h, 0cch, 0e0h, 070h, 01ch, 0cch, 078h, 000h
166 00016341 CCE0701CCC7800      <1>
167 00016348 FCB4303030307800CC- <1>     db  0fch, 0b4h, 030h, 030h, 030h, 030h, 078h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 0fch, 000h
167 00016351 CCCCCCCCCCFC00      <1>
168 00016358 CCCCCCCCCC783000C6- <1>     db  0cch, 0cch, 0cch, 0cch, 0cch, 078h, 030h, 000h, 0c6h, 0c6h, 0c6h, 0d6h, 0feh, 0eeh, 0c6h, 000h
168 00016361 C6C6D6FEEEC600      <1>
169 00016368 C6C66C38386CC600CC- <1>     db  0c6h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 030h, 078h, 000h
169 00016371 CCCC7830307800      <1>
170 00016378 FEC68C183266FE0078- <1>     db  0feh, 0c6h, 08ch, 018h, 032h, 066h, 0feh, 000h, 078h, 060h, 060h, 060h, 060h, 060h, 078h, 000h
170 00016381 60606060607800      <1>
171 00016388 C06030180C06020078- <1>     db  0c0h, 060h, 030h, 018h, 00ch, 006h, 002h, 000h, 078h, 018h, 018h, 018h, 018h, 018h, 078h, 000h
171 00016391 18181818187800      <1>
172 00016398 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
172 000163A1 000000000000FF      <1>
173 000163A8 3030180000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 076h, 000h
173 000163B1 00780C7CCC7600      <1>
174 000163B8 E060607C6666DC0000- <1>     db  0e0h, 060h, 060h, 07ch, 066h, 066h, 0dch, 000h, 000h, 000h, 078h, 0cch, 0c0h, 0cch, 078h, 000h
174 000163C1 0078CCC0CC7800      <1>
175 000163C8 1C0C0C7CCCCC760000- <1>     db  01ch, 00ch, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
175 000163D1 0078CCFCC07800      <1>
176 000163D8 386C60F06060F00000- <1>     db  038h, 06ch, 060h, 0f0h, 060h, 060h, 0f0h, 000h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 0f8h
176 000163E1 0076CCCC7C0CF8      <1>
177 000163E8 E0606C766666E60030- <1>     db  0e0h, 060h, 06ch, 076h, 066h, 066h, 0e6h, 000h, 030h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
177 000163F1 00703030307800      <1>
178 000163F8 0C000C0C0CCCCC78E0- <1>     db  00ch, 000h, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 0e0h, 060h, 066h, 06ch, 078h, 06ch, 0e6h, 000h
178 00016401 60666C786CE600      <1>
179 00016408 7030303030307800000- <1>     db  070h, 030h, 030h, 030h, 030h, 030h, 078h, 000h, 000h, 000h, 0cch, 0feh, 0feh, 0d6h, 0c6h, 000h
179 00016411 00CCFEFED6C600      <1>
180 00016418 0000F8CCCCCCCC0000- <1>     db  000h, 000h, 0f8h, 0cch, 0cch, 0cch, 0cch, 000h, 000h, 000h, 078h, 0cch, 0cch, 0cch, 078h, 000h
180 00016421 0078CCCCCC7800      <1>
181 00016428 0000DC66667C60F000- <1>     db  000h, 000h, 0dch, 066h, 066h, 07ch, 060h, 0f0h, 000h, 000h, 076h, 0cch, 0cch, 07ch, 00ch, 01eh
181 00016431 0076CCCC7C0C1E      <1>
182 00016438 0000DC766660F00000- <1>     db  000h, 000h, 0dch, 076h, 066h, 060h, 0f0h, 000h, 000h, 000h, 07ch, 0c0h, 078h, 00ch, 0f8h, 000h
```

```
182 00016441 007CC0780CF800      <1>
183 00016448 10307C303034180000- <1>     db  010h, 030h, 07ch, 030h, 030h, 034h, 018h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 076h, 000h
183 00016451 00CCCCCCCC7600      <1>
184 00016458 0000CCCCCC78300000- <1>     db  000h, 000h, 0cch, 0cch, 0cch, 078h, 030h, 000h, 000h, 000h, 0c6h, 0d6h, 0feh, 0feh, 06ch, 000h
184 00016461 00C6D6FEFE6C00      <1>
185 00016468 0000C66C386CC60000- <1>     db  000h, 000h, 0c6h, 06ch, 038h, 06ch, 0c6h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 07ch, 00ch, 0f8h
185 00016471 00CCCCCC7C0CF8      <1>
186 00016478 0000FC983064FC001C- <1>     db  000h, 000h, 0fch, 098h, 030h, 064h, 0fch, 000h, 01ch, 030h, 030h, 0e0h, 030h, 030h, 01ch, 000h
186 00016481 3030E030301C00      <1>
187 00016488 18181800181818E0- <1>     db  018h, 018h, 018h, 000h, 018h, 018h, 018h, 000h, 0e0h, 030h, 030h, 01ch, 030h, 030h, 0e0h, 000h
187 00016491 30301C3030E000      <1>
188 00016498 76DC00000000000000- <1>     db  076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h
188 000164A1 10386CC6C6FE00      <1>
189 000164A8 78CCC0CC78180C7800- <1>     db  078h, 0cch, 0c0h, 0cch, 078h, 018h, 00ch, 078h, 000h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
189 000164B1 CC00CCCCCC7E00      <1>
190 000164B8 1C0078CCFCC078007E- <1>     db  01ch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 07eh, 0c3h, 03ch, 006h, 03eh, 066h, 03fh, 000h
190 000164C1 C33C063E663F00      <1>
191 000164C8 CC00780C7CCC7E00E0- <1>     db  0cch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 0e0h, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h
191 000164D1 00780C7CCC7E00      <1>
192 000164D8 3030780C7CCC7E0000- <1>     db  030h, 030h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 000h, 000h, 078h, 0c0h, 0c0h, 078h, 00ch, 038h
192 000164E1 0078C0C0780C38      <1>
193 000164E8 7EC33C667E603C00CC- <1>     db  07eh, 0c3h, 03ch, 066h, 07eh, 060h, 03ch, 000h, 0cch, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h
193 000164F1 0078CCFCC07800      <1>
194 000164F8 E00078CCFCC07800CC- <1>     db  0e0h, 000h, 078h, 0cch, 0fch, 0c0h, 078h, 000h, 0cch, 000h, 070h, 030h, 030h, 030h, 078h, 000h
194 00016501 00703030307800      <1>
195 00016508 7CC6381818183C00E0- <1>     db  07ch, 0c6h, 038h, 018h, 018h, 018h, 03ch, 000h, 0e0h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
195 00016511 00703030307800      <1>
196 00016518 C6386CC6FEC6C60030- <1>     db  0c6h, 038h, 06ch, 0c6h, 0feh, 0c6h, 0c6h, 000h, 030h, 030h, 000h, 078h, 0cch, 0fch, 0cch, 000h
196 00016521 300078CCFCCC00      <1>
197 00016528 1C00FC607860FC0000- <1>     db  01ch, 000h, 0fch, 060h, 078h, 060h, 0fch, 000h, 000h, 000h, 07fh, 00ch, 07fh, 0cch, 07fh, 000h
197 00016531 007F0C7FCC7F00      <1>
198 00016538 3E6CCCFECCCCCE0078- <1>     db  03eh, 06ch, 0cch, 0feh, 0cch, 0cch, 0ceh, 000h, 078h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h
198 00016541 CC0078CCCC7800      <1>
199 00016548 00CC0078CCCC780000- <1>     db  000h, 0cch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 0e0h, 000h, 078h, 0cch, 0cch, 078h, 000h
199 00016551 E00078CCCC7800      <1>
200 00016558 78CC00CCCCCC7E0000- <1>     db  078h, 0cch, 000h, 0cch, 0cch, 0cch, 07eh, 000h, 000h, 0e0h, 000h, 0cch, 0cch, 0cch, 07eh, 000h
200 00016561 E000CCCCCC7E00      <1>
201 00016568 00CC00CCCC7C0CF8C3- <1>     db  000h, 0cch, 000h, 0cch, 0cch, 07ch, 00ch, 0f8h, 0c3h, 018h, 03ch, 066h, 066h, 03ch, 018h, 000h
201 00016571 183C66663C1800      <1>
202 00016578 CC00CCCCCCCC780018- <1>     db  0cch, 000h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 018h, 018h, 07eh, 0c0h, 0c0h, 07eh, 018h, 018h
202 00016581 187EC0C07E1818      <1>
203 00016588 386C64F060E6FC00CC- <1>     db  038h, 06ch, 064h, 0f0h, 060h, 0e6h, 0fch, 000h, 0cch, 0cch, 078h, 0fch, 030h, 0fch, 030h, 030h
203 00016591 CC78FC30FC3030      <1>
204 00016598 F8CCCCFAC6CFC6C70E- <1>     db  0f8h, 0cch, 0cch, 0fah, 0c6h, 0cfh, 0c6h, 0c7h, 00eh, 01bh, 018h, 03ch, 018h, 018h, 0d8h, 070h
204 000165A1 1B183C1818D870      <1>
205 000165A8 1C00780C7CCC7E0038- <1>     db  01ch, 000h, 078h, 00ch, 07ch, 0cch, 07eh, 000h, 038h, 000h, 070h, 030h, 030h, 030h, 078h, 000h
205 000165B1 00703030307800      <1>
206 000165B8 001C0078CCCC780000- <1>     db  000h, 01ch, 000h, 078h, 0cch, 0cch, 078h, 000h, 000h, 01ch, 000h, 0cch, 0cch, 0cch, 07eh, 000h
206 000165C1 1C00CCCCCC7E00      <1>
207 000165C8 00F800F8CCCCCC00FC- <1>     db  000h, 0f8h, 000h, 0f8h, 0cch, 0cch, 0cch, 000h, 0fch, 000h, 0cch, 0ech, 0fch, 0dch, 0cch, 000h
207 000165D1 00CCECFCDCCC00      <1>
208 000165D8 3C6C6C3E007E000038- <1>     db  03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h
208 000165E1 6C6C38007C0000      <1>
209 000165E8 30003060C0CC780000- <1>     db  030h, 000h, 030h, 060h, 0c0h, 0cch, 078h, 000h, 000h, 000h, 000h, 0fch, 0c0h, 0c0h, 000h, 000h
209 000165F1 0000FCC0C00000      <1>
210 000165F8 000000FC0C0C0000C3- <1>     db  000h, 000h, 000h, 0fch, 00ch, 00ch, 000h, 000h, 0c3h, 0c6h, 0cch, 0deh, 033h, 066h, 0cch, 00fh
210 00016601 C6CCDE3366CC0F      <1>
211 00016608 C3C6CCDB376FCF0318- <1>     db  0c3h, 0c6h, 0cch, 0dbh, 037h, 06fh, 0cfh, 003h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 000h
211 00016611 18001818181800      <1>
212 00016618 003366CC6633000000- <1>     db  000h, 033h, 066h, 0cch, 066h, 033h, 000h, 000h, 000h, 0cch, 066h, 033h, 066h, 0cch, 000h, 000h
212 00016621 CC663366CC0000      <1>
213 00016628 228822882288228855- <1>     db  022h, 088h, 022h, 088h, 022h, 088h, 022h, 088h, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
213 00016631 AA55AA55AA55AA      <1>
214 00016638 DB77DBEEDB77DBEE18- <1>     db  0dbh, 077h, 0dbh, 0eeh, 0dbh, 077h, 0dbh, 0eeh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
214 00016641 18181818181818      <1>
215 00016648 18181818F818181818- <1>     db  018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h
215 00016651 18F818F8181818      <1>
216 00016658 36363636F636363600- <1>     db  036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h
216 00016661 000000FE363636      <1>
217 00016668 0000F818F818181836- <1>     db  000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h
217 00016671 36F606F6363636      <1>
218 00016678 3636363636363600- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h
218 00016681 00FE06F6363636      <1>
219 00016688 3636F606FE00000036- <1>     db  036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h
219 00016691 363636FE000000      <1>
220 00016698 1818F818F800000000- <1>     db  018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h
220 000166A1 000000F8181818      <1>
221 000166A8 181818181F00000018- <1>     db  018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h
221 000166B1 181818FF000000      <1>
222 000166B8 00000000FF18181818- <1>     db  000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h
222 000166C1 1818181F181818      <1>
223 000166C8 00000000FF00000018- <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h
223 000166D1 181818FF181818      <1>
224 000166D8 18181F181F18181836- <1>     db  018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h
224 000166E1 36363637363636      <1>
225 000166E8 363637303F00000000- <1>     db  036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h
225 000166F1 003F3037363636      <1>
226 000166F8 3636F700FF00000000- <1>     db  036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h
226 00016701 00FF00F7363636      <1>
227 00016708 3636373037363636 00- <1>     db  036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
227 00016711 00FF00FF000000      <1>
228 00016718 3636F700F736363618- <1>     db  036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h
228 00016721 18FF00FF000000      <1>
229 00016728 36363636FF00000000- <1>     db  036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h
229 00016731 00FF00FF181818      <1>
230 00016738 00000000FF36363636- <1>     db  000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h
230 00016741 3636363F000000      <1>
231 00016748 18181F181F00000000- <1>     db  018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h
231 00016751 001F181F181818      <1>
232 00016758 000000003F36363636- <1>     db  000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h
232 00016761 363636FF363636      <1>
233 00016768 1818FF18FF18181818- <1>     db  018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h
233 00016771 181818F8000000      <1>
234 00016778 000000001F181818FF- <1>     db  000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
234 00016781 FFFFFFFFFFFFFF      <1>
235 00016788 00000000FFFFFFFFF0- <1>     db  000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
235 00016791 F0F0F0F0F0F0F0      <1>
236 00016798 0F0F0F0F0F0F0F0FFF- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h
236 000167A1 FFFFFF00000000      <1>
237 000167A8 000076DCC8DC760000- <1>     db  000h, 000h, 076h, 0dch, 0c8h, 0dch, 076h, 000h, 000h, 078h, 0cch, 0f8h, 0cch, 0f8h, 0c0h, 0c0h
237 000167B1 78CCF8CCF8C0C0      <1>
```

```
238 000167B8 00FCCCC0C0C0C00000- <1>     db  000h, 0fch, 0cch, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
238 000167C1 FE6C6C6C6C6C00      <1>
239 000167C8 FCCC603060CCFC0000- <1>     db  0fch, 0cch, 060h, 030h, 060h, 0cch, 0fch, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 070h, 000h
239 000167D1 007ED8D8D87000      <1>
240 000167D8 00666666667C60C000- <1>     db  000h, 066h, 066h, 066h, 066h, 07ch, 060h, 0c0h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 000h
240 000167E1 76DC1818181800      <1>
241 000167E8 FC3078CCCC7830FC38- <1>     db  0fch, 030h, 078h, 0cch, 0cch, 078h, 030h, 0fch, 038h, 06ch, 0c6h, 0feh, 0c6h, 06ch, 038h, 000h
241 000167F1 6CC6FEC66C3800      <1>
242 000167F8 386CC6C66C6CEE001C- <1>     db  038h, 06ch, 0c6h, 0c6h, 06ch, 06ch, 0eeh, 000h, 01ch, 030h, 018h, 07ch, 0cch, 0cch, 078h, 000h
242 00016801 30187CCCCC7800      <1>
243 00016808 00007EDBDB7E000006- <1>     db  000h, 000h, 07eh, 0dbh, 0dbh, 07eh, 000h, 000h, 006h, 00ch, 07eh, 0dbh, 0dbh, 07eh, 060h, 0c0h
243 00016811 0C7EDBDB7E60C0      <1>
244 00016818 3860C0F8C060380078- <1>     db  038h, 060h, 0c0h, 0f8h, 0c0h, 060h, 038h, 000h, 078h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 000h
244 00016821 CCCCCCCCCCCC00      <1>
245 00016828 00FC00FC00FC000030- <1>     db  000h, 0fch, 000h, 0fch, 000h, 0fch, 000h, 000h, 030h, 030h, 0fch, 030h, 030h, 000h, 0fch, 000h
245 00016831 30FC303000FC00      <1>
246 00016838 603018306000FC0018- <1>     db  060h, 030h, 018h, 030h, 060h, 000h, 0fch, 000h, 018h, 030h, 060h, 030h, 018h, 000h, 0fch, 000h
246 00016841 3060301800FC00      <1>
247 00016848 0E1B1B181818181818- <1>     db  00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 070h
247 00016851 18181818D8D870      <1>
248 00016858 303000FC0030300000- <1>     db  030h, 030h, 000h, 0fch, 000h, 030h, 030h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h
248 00016861 76DC0076DC0000      <1>
249 00016868 386C6C3800000000000- <1>    db  038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h
249 00016871 00001818000000      <1>
250 00016878 000000001800000000F- <1>    db  000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 0ech, 06ch, 03ch, 01ch
250 00016881 0C0C0CEC6C3C1C      <1>
251 00016888 786C6C6C6C00000070- <1>     db  078h, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 070h, 018h, 030h, 060h, 078h, 000h, 000h, 000h
251 00016891 18306078000000      <1>
252 00016898 00003C3C3C3C000000- <1>     db  000h, 000h, 03ch, 03ch, 03ch, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
252 000168A1 00000000000000      <1>
253                              <1> vgafont14:
254 000168A8 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
254 000168B1 00000000000000      <1>
255 000168B8 7E81A58181BD99817E- <1>     db  07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 07eh, 000h, 000h, 000h, 000h, 000h, 07eh, 0ffh
255 000168C1 00000000007EFF      <1>
256 000168C8 DBFFFFC3E7FF7E0000- <1>     db  0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 0feh, 0feh
256 000168D1 000000006CFEFE      <1>
257 000168D8 FEFE7C3810000000000- <1>    db  0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch
257 000168E1 000010387CFE7C      <1>
258 000168E8 3810000000000000018- <1>    db  038h, 010h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h
258 000168F1 3C3CE7E7E71818      <1>
259 000168F8 3C0000000000183C7E- <1>     db  03ch, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h
259 00016901 FFFF7E18183C00      <1>
260 00016908 00000000000000183C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
260 00016911 3C180000000000      <1>
261 00016918 FFFFFFFFFFE7C3C3E7- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h
261 00016921 FFFFFFFFFF0000      <1>
262 00016928 00003C664242663C00- <1>     db  000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh
262 00016931 000000FFFFFFFF      <1>
263 00016938 C399BDBD99C3FFFFFF- <1>     db  0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 01eh, 00eh, 01ah, 032h
263 00016941 FF00001E0E1A32      <1>
264 00016948 78CCCCCC7800000000- <1>     db  078h, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 066h, 066h, 03ch, 018h
264 00016951 003C6666663C18      <1>
265 00016958 7E1818000000000003F- <1>    db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 070h, 0f0h
265 00016961 333F30303070F0      <1>
266 00016968 E000000000007F637F- <1>     db  0e0h, 000h, 000h, 000h, 000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h
266 00016971 63636367E7E6C0      <1>
267 00016978 000000001818DB3CE7- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h
267 00016981 3CDB1818000000      <1>
268 00016988 000080C0E0F8FEF8E0- <1>     db  000h, 000h, 080h, 0c0h, 0e0h, 0f8h, 0feh, 0f8h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h, 000h
268 00016991 C0800000000000      <1>
269 00016998 02060E3EFE3E0E0602- <1>     db  002h, 006h, 00eh, 03eh, 0feh, 03eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch
269 000169A1 0000000000183C      <1>
270 000169A8 7E1818187E3C180000- <1>     db  07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h
270 000169B1 00000066666666      <1>
271 000169B8 666600666660000000- <1>     db  066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh
271 000169C1 007FDBDBDB7B1B      <1>
272 000169C8 1B1B1B000000007CC6- <1>     db  01bh, 01bh, 01bh, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h
272 000169D1 60386CC6C66C38      <1>
273 000169D8 0CC67C000000000000- <1>     db  00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 000h
273 000169E1 000000FEFEFE00      <1>
274 000169E8 00000000183C7E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h
274 000169F1 187E3C187E0000      <1>
275 000169F8 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
275 00016A01 18180000000000      <1>
276 00016A08 1818181818187E3C18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
276 00016A11 00000000000000      <1>
277 00016A18 180CFE0C1800000000- <1>     db  018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 030h, 060h
277 00016A21 00000000003060      <1>
278 00016A28 FE6030000000000000- <1>     db  0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h
278 00016A31 00000000C0C0C0      <1>
279 00016A38 FE00000000000000000- <1>    db  0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 028h, 06ch, 0feh, 06ch, 028h, 000h
279 00016A41 00286CFE6C2800      <1>
280 00016A48 000000000000001038- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h
280 00016A51 387C7CFEFE0000      <1>
281 00016A58 0000000000FEFE7C7C- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h
281 00016A61 38381000000000      <1>
282 00016A68 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
282 00016A71 00000000000000      <1>
283 00016A78 183C3C3C1818001818- <1>     db  018h, 03ch, 03ch, 03ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 066h, 066h, 066h
283 00016A81 00000000666666      <1>
284 00016A88 240000000000000000- <1>     db  024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch
284 00016A91 0000006C6CFE6C      <1>
285 00016A98 6C6CFE6C6C00000018- <1>     db  06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h
285 00016AA1 187CC6C2C07C06      <1>
286 00016AA8 86C67C181800000000- <1>     db  086h, 0c6h, 07ch, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 066h
286 00016AB1 00C2C60C183066      <1>
287 00016AB8 C600000000000386C6C- <1>    db  0c6h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 076h, 000h
287 00016AC1 3876DCCCCC7600      <1>
288 00016AC8 000003030306000000- <1>     db  000h, 000h, 000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
288 00016AD1 00000000000000      <1>
289 00016AD8 00000C1830303030- <1>       db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h, 000h
289 00016AE1 180C0000000000      <1>
290 00016AE8 30180C0C0C0C0C1830- <1>     db  030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
290 00016AF1 00000000000000      <1>
291 00016AF8 663CFF3C66000000000- <1>    db  066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
291 00016B01 00000000001818      <1>
292 00016B08 7E18180000000000000- <1>    db  07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
292 00016B11 00000000000000      <1>
293 00016B18 1818183000000000000- <1>    db  018h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h
293 00016B21 000000FE000000      <1>
```

```
294 00016B28 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
294 00016B31 00000000181800      <1>
295 00016B38 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
295 00016B41 60C08000000000      <1>
296 00016B48 00007CC6CEDEF6E6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0ceh, 0deh, 0f6h, 0e6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
296 00016B51 C67C0000000000      <1>
297 00016B58 183878181818187E- <1>     db  018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h
297 00016B61 00000000007CC6      <1>
298 00016B68 060C183060C6FE0000- <1>     db  006h, 00ch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 006h, 006h
298 00016B71 0000007CC60606      <1>
299 00016B78 3C0606C67C00000000- <1>     db  03ch, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh
299 00016B81 000C1C3C6CCCFE      <1>
300 00016B88 0C0C1E0000000000FE- <1>     db  00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 0c6h
300 00016B91 C0C0C0FC0606C6      <1>
301 00016B98 7C000000000003860C0- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 07ch, 000h
301 00016BA1 C0FCC6C6C67C00      <1>
302 00016BA8 00000000FEC6060C18- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c6h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h
302 00016BB1 30303030000000      <1>
303 00016BB8 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
303 00016BC1 C67C0000000000      <1>
304 00016BC8 7CC6C6C67E06060C78- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h, 000h, 000h, 018h
304 00016BD1 00000000000018      <1>
305 00016BD8 1800000018180000000- <1>     db  018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h
305 00016BE1 00000000181800      <1>
306 00016BE8 000018183000000000- <1>     db  000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h
306 00016BF1 00060C18306030      <1>
307 00016BF8 180C060000000000000- <1>     db  018h, 00ch, 006h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h
307 00016C01 00007E00007E00      <1>
308 00016C08 000000000000603018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h
308 00016C11 0C060C18306000      <1>
309 00016C18 000000007CC6C60C18- <1>     db  000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h
309 00016C21 18001818000000      <1>
310 00016C28 00007CC6C6DEDEDEDC- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h, 000h
310 00016C31 C07C0000000000      <1>
311 00016C38 10386CC6C6FEC6C6C6- <1>     db  010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h, 0fch, 066h
311 00016C41 0000000000FC66      <1>
312 00016C48 66667C666666FC0000- <1>     db  066h, 066h, 07ch, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h
312 00016C51 0000003C66C2C0      <1>
313 00016C58 C0C0C2663C00000000- <1>     db  0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h
313 00016C61 00F86C66666666      <1>
314 00016C68 666CF800000000000FE- <1>     db  066h, 06ch, 0f8h, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 062h, 066h
314 00016C71 66626878686266      <1>
315 00016C78 FE0000000000FE6662- <1>     db  0feh, 000h, 000h, 000h, 000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 0f0h, 000h
315 00016C81 6878686060F000      <1>
316 00016C88 000000003C66C2C0C0- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 066h, 03ah, 000h, 000h, 000h
316 00016C91 DEC6663A000000      <1>
317 00016C98 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
317 00016CA1 C6C600000000000     <1>
318 00016CA8 3C1818181818183C- <1>     db  03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 01eh, 00ch
318 00016CB1 00000000001E0C      <1>
319 00016CB8 0C0C0C0CCCCC780000- <1>     db  00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h, 000h, 0e6h, 066h, 06ch, 06ch
319 00016CC1 000000E6666C6C      <1>
320 00016CC8 786C6C66E600000000- <1>     db  078h, 06ch, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h
320 00016CD1 00F06060606060      <1>
321 00016CD8 6266FE0000000000C6- <1>     db  062h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 0c6h, 0eeh, 0feh, 0feh, 0d6h, 0c6h, 0c6h, 0c6h
321 00016CE1 EEFEFED6C6C6C6      <1>
322 00016CE8 C600000000000C6E6F6- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h
322 00016CF1 FEDECEC6C6C600      <1>
323 00016CF8 00000000386CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h
323 00016D01 C6C66C38000000      <1>
324 00016D08 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h
324 00016D11 60F00000000000      <1>
325 00016D18 7CC6C6C6C6D6DE7C0C- <1>     db  07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h, 000h, 000h, 0fch, 066h
325 00016D21 0E000000000FC66     <1>
326 00016D28 66667C6C6666E60000- <1>     db  066h, 066h, 07ch, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 060h
326 00016D31 0000007CC6C660      <1>
327 00016D38 380CC6C67C00000000- <1>     db  038h, 00ch, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 07eh, 07eh, 05ah, 018h, 018h, 018h
327 00016D41 007E7E5A181818      <1>
328 00016D48 18183C0000000000C6- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h
328 00016D51 C6C6C6C6C6C6C6      <1>
329 00016D58 7C00000000000C6C6C6- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 010h, 000h
329 00016D61 C6C6C66C381000      <1>
330 00016D68 00000000C6C6C6C6D6- <1>     db  000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 07ch, 06ch, 000h, 000h, 000h
330 00016D71 D6FE7C6C000000      <1>
331 00016D78 0000C6C66C3838386C- <1>     db  000h, 000h, 0c6h, 0c6h, 06ch, 038h, 038h, 038h, 06ch, 0c6h, 0c6h, 000h, 000h, 000h, 000h, 000h
331 00016D81 C6C600000000000     <1>
332 00016D88 666666663C1818183C- <1>     db  066h, 066h, 066h, 066h, 03ch, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h
332 00016D91 0000000000FEC6      <1>
333 00016D98 8C183060C2C6FE0000- <1>     db  08ch, 018h, 030h, 060h, 0c2h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 03ch, 030h, 030h, 030h
333 00016DA1 0000003C303030      <1>
334 00016DA8 303030303C00000000- <1>     db  030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch
334 00016DB1 0080C0E070381C      <1>
335 00016DB8 0E060200000000003C- <1>     db  00eh, 006h, 002h, 000h, 000h, 000h, 000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch
335 00016DC1 0C0C0C0C0C0C0C      <1>
336 00016DC8 3C00000010386CC600- <1>     db  03ch, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
336 00016DD1 00000000000000      <1>
337 00016DD8 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h
337 00016DE1 0000000000FF00      <1>
338 00016DE8 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
338 00016DF1 00000000000000      <1>
339 00016DF8 000000780C7CCCCC76- <1>     db  000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0e0h, 060h
339 00016E01 0000000000E060      <1>
340 00016E08 60786C6666667C0000- <1>     db  060h, 078h, 06ch, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
340 00016E11 0000000000007C      <1>
341 00016E18 C6C0C0C67C00000000- <1>     db  0c6h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch
341 00016E21 001C0C0C3C6CCC      <1>
342 00016E28 CCCC760000000000000- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h
342 00016E31 00007CC6FEC0C6      <1>
343 00016E38 7C0000000000386C64- <1>     db  07ch, 000h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 0f0h, 000h
343 00016E41 60F0606060F000      <1>
344 00016E48 0000000000000076CC- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
344 00016E51 CCCC7C0CCC7800      <1>
345 00016E58 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h
345 00016E61 66E60000000000      <1>
346 00016E68 1818003818181818183C- <1>     db  018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 006h, 006h
346 00016E71 00000000000606      <1>
347 00016E78 000E06060606666663C- <1>     db  000h, 00eh, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h, 000h, 000h, 0e0h, 060h, 060h, 066h
347 00016E81 000000E0606066      <1>
348 00016E88 6C786C66E600000000- <1>     db  06ch, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h
348 00016E91 00381818181818      <1>
349 00016E98 18183C0000000000000- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ech, 0feh, 0d6h, 0d6h, 0d6h
```

```
349 00016EA1 0000ECFED6D6D6      <1>
350 00016EA8 C600000000000000000- <1>     db  0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 000h
350 00016EB1 DC666666666600      <1>
351 00016EB8 000000000000007CC6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h
351 00016EC1 C6C6C67C000000      <1>
352 00016EC8 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h, 000h, 000h
352 00016ED1 7C6060F0000000      <1>
353 00016ED8 00000076CCCCCC7C0C- <1>     db  000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h, 000h
353 00016EE1 0C1E0000000000      <1>
354 00016EE8 00DC76666060F00000- <1>     db  000h, 0dch, 076h, 066h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch
354 00016EF1 0000000000007C      <1>
355 00016EF8 C6701CC67C00000000- <1>     db  0c6h, 070h, 01ch, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h
355 00016F01 00103030FC3030      <1>
356 00016F08 30361C000000000000- <1>     db  030h, 036h, 01ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch
356 00016F11 0000CCCCCCCCCC      <1>
357 00016F18 760000000000000000- <1>     db  076h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 03ch, 018h, 000h
357 00016F21 666666663C1800      <1>
358 00016F28 000000000000C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0d6h, 0d6h, 0feh, 06ch, 000h, 000h, 000h
358 00016F31 D6D6FE6C000000      <1>
359 00016F38 0000000000C66C3838- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 06ch, 038h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h
359 00016F41 6CC60000000000      <1>
360 00016F48 000000C6C6C6C67E06- <1>     db  000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h, 000h, 000h, 000h, 000h
360 00016F51 0CF80000000000      <1>
361 00016F58 00FECC183066FE0000- <1>     db  000h, 0feh, 0cch, 018h, 030h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h, 00eh, 018h, 018h, 018h
361 00016F61 0000000E181818      <1>
362 00016F68 701818180E00000000- <1>     db  070h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h
362 00016F71 00181818180018      <1>
363 00016F78 1818180000000000070- <1>     db  018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h
363 00016F81 1818180E181818      <1>
364 00016F88 700000000000076DC00- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
364 00016F91 00000000000000      <1>
365 00016F98 0000000000010386C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h
365 00016FA1 C6C6FE00000000      <1>
366 00016FA8 00003C66C2C0C0C266- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h, 000h
366 00016FB1 3C0C067C000000      <1>
367 00016FB8 CCCC00CCCCCCCCCC76- <1>     db  0cch, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h
367 00016FC1 000000000C1830      <1>
368 00016FC8 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 078h
368 00016FD1 000010386C0078      <1>
369 00016FD8 0C7CCCCC7600000000- <1>     db  00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 000h, 078h, 00ch, 07ch
369 00016FE1 00CCCC00780C7C      <1>
370 00016FE8 CCCC76000000006030- <1>     db  0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch
370 00016FF1 1800780C7CCCCC      <1>
371 00016FF8 7600000000386C3800- <1>     db  076h, 000h, 000h, 000h, 000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h
371 00017001 780C7CCCCC7600      <1>
372 00017008 0000000000003C6660- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 03ch, 066h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h
372 00017011 663C0C063C0000      <1>
373 00017018 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
373 00017021 C67C0000000000      <1>
374 00017028 CCCC007CC6FEC0C67C- <1>     db  0cch, 0cch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h
374 00017031 00000000603018      <1>
375 00017038 007CC6FEC0C67C0000- <1>     db  000h, 07ch, 0c6h, 0feh, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 000h, 038h
375 00017041 000000066660038     <1>
376 00017048 181818183C00000000- <1>     db  018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h
376 00017051 183C6600381818      <1>
377 00017058 18183C000000006030- <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h
377 00017061 18003818181818      <1>
378 00017068 3C00000000C6C61038- <1>     db  03ch, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h
378 00017071 6CC6C6FEC6C600      <1>
379 00017078 0000386C3800386CC6- <1>     db  000h, 000h, 038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 000h, 000h, 000h
379 00017081 C6FEC6C6000000      <1>
380 00017088 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 066h, 0feh, 000h, 000h, 000h, 000h, 000h
380 00017091 66FE0000000000      <1>
381 00017098 0000CC76367ED8D86E- <1>     db  000h, 000h, 0cch, 076h, 036h, 07eh, 0d8h, 0d8h, 06eh, 000h, 000h, 000h, 000h, 000h, 03eh, 06ch
381 000170A1 000000000003E6C     <1>
382 000170A8 CCCCFECCCCCCCCE0000- <1>     db  0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h, 010h, 038h, 06ch, 000h, 07ch
382 000170B1 000010386C007C      <1>
383 000170B8 C6C6C6C67C00000000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h, 07ch, 0c6h, 0c6h
383 000170C1 00C6C6007CC6C6      <1>
384 000170C8 C6C67C000000006030- <1>     db  0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h
384 000170D1 18007CC6C6C6C6      <1>
385 000170D8 7C0000000003078CC00- <1>     db  07ch, 000h, 000h, 000h, 000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h
385 000170E1 CCCCCCCCCC7600      <1>
386 000170E8 000000603018000CCCC- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h
386 000170F1 CCCCCC76000000      <1>
387 000170F8 0000C6C600C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h, 000h, 0c6h
387 00017101 7E060C780000C6      <1>
388 00017108 C6386CC6C6C6C66C38- <1>     db  0c6h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 000h
388 00017111 00000000C6C600      <1>
389 00017118 C6C6C6C6C6C67C0000- <1>     db  0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 018h, 03ch, 066h, 060h
389 00017121 000018183C6660      <1>
390 00017128 60663C1818000000000- <1>     db  060h, 066h, 03ch, 018h, 018h, 000h, 000h, 000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h
390 00017131 386C6460F06060      <1>
391 00017138 60E6FC000000000066- <1>     db  060h, 0e6h, 0fch, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 03ch, 018h, 07eh, 018h, 07eh, 018h
391 00017141 663C187E187E18      <1>
392 00017148 1800000000F8CCCCF8- <1>     db  018h, 000h, 000h, 000h, 000h, 0f8h, 0cch, 0cch, 0f8h, 0c4h, 0cch, 0deh, 0cch, 0cch, 0c6h, 000h
392 00017151 C4CCDECCCCC600      <1>
393 00017158 0000000E1B1818187E- <1>     db  000h, 000h, 000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h
393 00017161 18181818D87000      <1>
394 00017168 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 00ch
394 00017171 CC76000000000C      <1>
395 00017178 183000381818183C- <1>     db  018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h
395 00017181 000000183060        <1>
396 00017188 007CC6C6C67C0000- <1>     db  000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 018h, 030h, 060h, 000h, 0cch
396 00017191 000018306000CC      <1>
397 00017198 CCCCCCCC7600000000- <1>     db  0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h
397 000171A1 0076DC00DC6666      <1>
398 000171A8 66666600000076DC00- <1>     db  066h, 066h, 066h, 000h, 000h, 000h, 076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h
398 000171B1 C6E6F6FEDECEC6      <1>
399 000171B8 C6000000003C6C6C3E- <1>     db  0c6h, 000h, 000h, 000h, 000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h
399 000171C1 007E0000000000      <1>
400 000171C8 000000386C6C38007C- <1>     db  000h, 000h, 000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h
400 000171D1 00000000000000      <1>
401 000171D8 0000303000303060C6- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h, 000h
401 000171E1 C67C0000000000      <1>
402 000171E8 00000000FEC0C0C000- <1>     db  000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
402 000171F1 00000000000000      <1>
403 000171F8 0000FE0606060000000- <1>     db  000h, 000h, 0feh, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h
403 00017201 0000C0C0C6CCD8      <1>
404 00017208 3060DC860C183E0000- <1>     db  030h, 060h, 0dch, 086h, 00ch, 018h, 03eh, 000h, 000h, 0c0h, 0c0h, 0c6h, 0cch, 0d8h, 030h, 066h
404 00017211 C0C0C6CCD83066      <1>
```

```
405 00017218 CE9E3E060600000018- <1>     db  0ceh, 09eh, 03eh, 006h, 006h, 000h, 000h, 000h, 018h, 018h, 000h, 018h, 018h, 03ch, 03ch, 03ch
405 00017221 180018183C3C3C      <1>
406 00017228 180000000000000036- <1>     db  018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h
406 00017231 6CD86C36000000      <1>
407 00017238 000000000000D86C36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h
407 00017241 6CD80000000000      <1>
408 00017248 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 055h, 0aah
408 00017251 441144114455AA      <1>
409 00017258 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 0ddh, 077h, 0ddh, 077h
409 00017261 AA55AADD77DD77      <1>
410 00017268 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 018h, 018h, 018h, 018h, 018h, 018h
410 00017271 77181818181818      <1>
411 00017278 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h
411 00017281 181818181818F8      <1>
412 00017288 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h
412 00017291 1818F818F81818      <1>
413 00017298 1818181836363636- <1>     db  018h, 018h, 018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h
413 000172A1 3636F636363636      <1>
414 000172A8 3636000000000000- <1>     db  036h, 036h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h
414 000172B1 FE363636363636      <1>
415 000172B8 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 036h, 036h
415 000172C1 18181818183636      <1>
416 000172C8 363636F606F6363636- <1>     db  036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
416 000172D1 36363636363636      <1>
417 000172D8 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0feh
417 000172E1 3600000000000FE      <1>
418 000172E8 06F6363636363636- <1>     db  006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh
418 000172F1 36363636F606FE      <1>
419 000172F8 0000000000003636- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h
419 00017301 36363636FE0000      <1>
420 00017308 0000000018181818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h
420 00017311 F818F800000000      <1>
421 00017318 0000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h
421 00017321 F8181818181818      <1>
422 00017328 1818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h
422 00017331 00000000001818      <1>
423 00017338 1818181818FF000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
423 00017341 00000000000000      <1>
424 00017348 000000FF1818181818- <1>     db  000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
424 00017351 18181818181818      <1>
425 00017358 181F18181818181800- <1>     db  018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
425 00017361 0000000000000FF      <1>
426 00017368 0000000000000181818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h
426 00017371 18181818FF1818      <1>
427 00017378 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h
427 00017381 1F181F18181818      <1>
428 00017388 1818363636363636- <1>     db  018h, 018h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h
428 00017391 37363636363636      <1>
429 00017398 3636363637303F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
429 000173A1 00000000000000      <1>
430 000173A8 0000003F3037363636- <1>     db  000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
430 000173B1 36363636363636      <1>
431 000173B8 36F700FF00000000000- <1>     db  036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
431 000173C1 0000000000000FF      <1>
432 000173C8 00F7363636363636- <1>     db  000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h
432 000173D1 36363636373037      <1>
433 000173D8 3636363636000000- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h
433 000173E1 0000FF00FF0000      <1>
434 000173E8 0000000036363636- <1>     db  000h, 000h, 000h, 000h, 036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h
434 000173F1 F700F736363636      <1>
435 000173F8 363618181818FF00- <1>     db  036h, 036h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h
435 00017401 FF000000000000      <1>
436 00017408 3636363636363FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
436 00017411 00000000000000      <1>
437 00017418 000000FF00FF181818- <1>     db  000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
437 00017421 18181800000000      <1>
438 00017428 000000FF3636363636- <1>     db  000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
438 00017431 36363636363636      <1>
439 00017438 363F000000000000018- <1>     db  036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh
439 00017441 181818181F181F      <1>
440 00017448 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h
440 00017451 00001F181F1818      <1>
441 00017458 1818181800000000000- <1>     db  018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h
441 00017461 00003F36363636      <1>
442 00017468 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h
442 00017471 FF363636363636      <1>
443 00017478 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
443 00017481 18181818181818      <1>
444 00017488 1818181818F8000000- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
444 00017491 00000000000000      <1>
445 00017498 0000001F1818181818- <1>     db  000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
445 000174A1 18FFFFFFFFFFFF      <1>
446 000174A8 FFFFFFFFFFFFFFFF00- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh
446 000174B1 0000000000000FF      <1>
447 000174B8 FFFFFFFFFFFFF0F0F0- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
447 000174C1 F0F0F0F0F0F0F0      <1>
448 000174C8 F0F0F0F00F0F0F0F0F- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
448 000174D1 0F0F0F0F0F0F0F      <1>
449 000174D8 0F0FFFFFFFFFFFFFFF- <1>     db  00fh, 00fh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
449 000174E1 00000000000000      <1>
450 000174E8 0000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h, 000h
450 000174F1 DC760000000000      <1>
451 000174F8 00007CC6FCC6C6FCC0- <1>     db  000h, 000h, 07ch, 0c6h, 0fch, 0c6h, 0c6h, 0fch, 0c0h, 0c0h, 040h, 000h, 000h, 000h, 0feh, 0c6h
451 00017501 C040000000FEC6      <1>
452 00017508 C6C0C0C0C0C0C00000- <1>     db  0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 06ch
452 00017511 0000000000FE6C      <1>
453 00017518 6C6C6C6C6C00000000- <1>     db  06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h
453 00017521 00FEC660301830      <1>
454 00017528 60C6FE000000000000- <1>     db  060h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h
454 00017531 00007ED8D8D8D8      <1>
455 00017538 7000000000000000066- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h
455 00017541 6666667C6060C0      <1>
456 00017548 00000000000076DC18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h
456 00017551 18181818000000      <1>
457 00017558 00007E183C6666663C- <1>     db  000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h, 000h
457 00017561 187E00000000000      <1>
458 00017568 386CC6C6FEC6C66C38- <1>     db  038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 038h, 06ch
458 00017571 0000000000386C      <1>
459 00017578 C6C6C66C6C6CEE0000- <1>     db  0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h, 000h, 01eh, 030h, 018h, 00ch
459 00017581 0000001E30180C      <1>
460 00017588 3E6666663C00000000- <1>     db  03eh, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh
```

```
00017591 000000007EDBDB      <1>
00017598 7E0000000000000003- <1>     db  07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h
000175A1 067EDBDBF37E60      <1>
000175A8 C000000000001C3060- <1>     db  0c0h, 000h, 000h, 000h, 000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 030h, 01ch, 000h
000175B1 607C6060301C00      <1>
000175B8 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h
000175C1 C6C6C6C6000000      <1>
000175C8 000000FE0000FE0000- <1>     db  000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
000175D1 FE000000000000      <1>
000175D8 0018187E18180000FF- <1>     db  000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 030h, 018h
000175E1 00000000003018      <1>
000175E8 0C060C1830007E0000- <1>     db  00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00ch, 018h, 030h, 060h
000175F1 0000000C183060      <1>
000175F8 30180C007E00000000- <1>     db  030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h
00017601 000E1B1B181818      <1>
00017608 1818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h
00017611 1818181818D8D8      <1>
00017618 700000000000001818- <1>     db  070h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h
00017621 007E0018180000      <1>
00017628 00000000000076DC00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h
00017631 76DC0000000000      <1>
00017638 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017641 00000000000000      <1>
00017648 0000000018180000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
00017651 00000000000000      <1>
00017658 0000001800000000000- <1>     db  000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 00fh, 00ch, 00ch, 00ch, 00ch
00017661 00000F0C0C0C0C      <1>
00017668 0CEC6C3C1C00000000- <1>     db  00ch, 0ech, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h
00017671 D86C6C6C6C6C00      <1>
00017678 0000000000000070D8- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h
00017681 3060C8F8000000      <1>
00017688 000000000000000007C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h
00017691 7C7C7C7C7C0000      <1>
00017698 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
000176A1 000000000000000     <1>
                             <1> vgafont16:
000176A8 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
000176B1 00000000000000      <1>
000176B8 00007E81A58181BD99- <1>     db  000h, 000h, 07eh, 081h, 0a5h, 081h, 081h, 0bdh, 099h, 081h, 081h, 07eh, 000h, 000h, 000h, 000h
000176C1 81817E00000000      <1>
000176C8 00007EFFDBFFFFC3E7- <1>     db  000h, 000h, 07eh, 0ffh, 0dbh, 0ffh, 0ffh, 0c3h, 0e7h, 0ffh, 0ffh, 07eh, 000h, 000h, 000h, 000h
000176D1 FFFF7E00000000      <1>
000176D8 000000006CFEFEFEFE- <1>     db  000h, 000h, 000h, 000h, 06ch, 0feh, 0feh, 0feh, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h
000176E1 7C3810000000000     <1>
000176E8 0000000010387CFE7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 07ch, 0feh, 07ch, 038h, 010h, 000h, 000h, 000h, 000h, 000h
000176F1 38100000000000      <1>
000176F8 000000183C3CE7E7E7- <1>     db  000h, 000h, 000h, 018h, 03ch, 03ch, 0e7h, 0e7h, 0e7h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
00017701 18183C00000000      <1>
00017708 000000183C7EFFFF7E- <1>     db  000h, 000h, 000h, 018h, 03ch, 07eh, 0ffh, 0ffh, 07eh, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
00017711 18183C00000000      <1>
00017718 000000000000183C3C- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 018h, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
00017721 18000000000000      <1>
00017728 FFFFFFFFFFFFE7C3C3- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0e7h, 0c3h, 0c3h, 0e7h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
00017731 E7FFFFFFFFFFFF      <1>
00017738 00000000003C664242- <1>     db  000h, 000h, 000h, 000h, 000h, 03ch, 066h, 042h, 042h, 066h, 03ch, 000h, 000h, 000h, 000h, 000h
00017741 663C0000000000      <1>
00017748 FFFFFFFFFFC399BDBD- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0c3h, 099h, 0bdh, 0bdh, 099h, 0c3h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
00017751 99C3FFFFFFFFFF      <1>
00017758 00001E0E1A3278CCCC- <1>     db  000h, 000h, 01eh, 00eh, 01ah, 032h, 078h, 0cch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
00017761 CCCC7800000000      <1>
00017768 00003C666666663C18- <1>     db  000h, 000h, 03ch, 066h, 066h, 066h, 066h, 03ch, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
00017771 7E181800000000      <1>
00017778 00003F333F30303030- <1>     db  000h, 000h, 03fh, 033h, 03fh, 030h, 030h, 030h, 030h, 070h, 0f0h, 0e0h, 000h, 000h, 000h, 000h
00017781 70F0E000000000      <1>
00017788 00007F637F63636363- <1>     db  000h, 000h, 07fh, 063h, 07fh, 063h, 063h, 063h, 063h, 067h, 0e7h, 0e6h, 0c0h, 000h, 000h, 000h
00017791 67E7E6C0000000      <1>
00017798 0000001818DB3CE73C- <1>     db  000h, 000h, 000h, 018h, 018h, 0dbh, 03ch, 0e7h, 03ch, 0dbh, 018h, 018h, 000h, 000h, 000h, 000h
000177A1 DB181800000000      <1>
000177A8 0080C0E0F0F8FEF8F0- <1>     db  000h, 080h, 0c0h, 0e0h, 0f0h, 0f8h, 0feh, 0f8h, 0f0h, 0e0h, 0c0h, 080h, 000h, 000h, 000h, 000h
000177B1 E0C08000000000      <1>
000177B8 0002060E1E3EFE3E1E- <1>     db  000h, 002h, 006h, 00eh, 01eh, 03eh, 0feh, 03eh, 01eh, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
000177C1 0E060200000000      <1>
000177C8 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h, 000h
000177D1 3C180000000000      <1>
000177D8 000066666666666666- <1>     db  000h, 000h, 066h, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 066h, 066h, 000h, 000h, 000h, 000h
000177E1 00666600000000      <1>
000177E8 00007FDBDBDB7B1B1B- <1>     db  000h, 000h, 07fh, 0dbh, 0dbh, 0dbh, 07bh, 01bh, 01bh, 01bh, 01bh, 01bh, 000h, 000h, 000h, 000h
000177F1 1B1B1B00000000      <1>
000177F8 007CC660386CC6C66C- <1>     db  000h, 07ch, 0c6h, 060h, 038h, 06ch, 0c6h, 0c6h, 06ch, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h
00017801 380CC67C000000      <1>
00017808 0000000000000000FE- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0feh, 0feh, 0feh, 000h, 000h, 000h, 000h
00017811 FEFEFE00000000      <1>
00017818 0000183C7E1818187E- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 07eh, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
00017821 3C187E00000000      <1>
00017828 0000183C7E18181818- <1>     db  000h, 000h, 018h, 03ch, 07eh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
00017831 18181800000000      <1>
00017838 0000181818181818 18- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 03ch, 018h, 000h, 000h, 000h, 000h
00017841 7E3C1800000000      <1>
00017848 0000000000180CFE0C- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 00ch, 0feh, 00ch, 018h, 000h, 000h, 000h, 000h, 000h, 000h
00017851 18000000000000      <1>
00017858 00000000003060FE60- <1>     db  000h, 000h, 000h, 000h, 000h, 030h, 060h, 0feh, 060h, 030h, 000h, 000h, 000h, 000h, 000h, 000h
00017861 30000000000000      <1>
00017868 0000000000000C0C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0c0h, 0c0h, 0c0h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h
00017871 FE000000000000      <1>
00017878 00000000002466FF66- <1>     db  000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h
00017881 240000000000000     <1>
00017888 000000001038387C7C- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 038h, 07ch, 07ch, 0feh, 0feh, 000h, 000h, 000h, 000h, 000h
00017891 FEFE0000000000      <1>
00017898 00000000FEFE7C7C38- <1>     db  000h, 000h, 000h, 000h, 0feh, 0feh, 07ch, 07ch, 038h, 038h, 010h, 000h, 000h, 000h, 000h, 000h
000178A1 38100000000000      <1>
000178A8 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
000178B1 00000000000000      <1>
000178B8 0000183C3C3C181818- <1>     db  000h, 000h, 018h, 03ch, 03ch, 03ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
000178C1 00181800000000      <1>
000178C8 0066666624000000000- <1>     db  000h, 066h, 066h, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
000178D1 00000000000000      <1>
000178D8 0000006C6CFE6C6C6C- <1>     db  000h, 000h, 000h, 06ch, 06ch, 0feh, 06ch, 06ch, 06ch, 0feh, 06ch, 06ch, 000h, 000h, 000h, 000h
000178E1 FE6C6C00000000      <1>
000178E8 18187CC6C2C07C0606- <1>     db  018h, 018h, 07ch, 0c6h, 0c2h, 0c0h, 07ch, 006h, 006h, 086h, 0c6h, 07ch, 018h, 018h, 000h, 000h
000178F1 86C67C18180000      <1>
000178F8 00000000C2C60C1830- <1>     db  000h, 000h, 000h, 000h, 0c2h, 0c6h, 00ch, 018h, 030h, 060h, 0c6h, 086h, 000h, 000h, 000h, 000h
```

```
516 00017901 60C68600000000      <1>
517 00017908 0000386C6C3876DCCC- <1>     db  000h, 000h, 038h, 06ch, 06ch, 038h, 076h, 0dch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
517 00017911 CCCC7600000000      <1>
518 00017918 0030303060000000000- <1>    db  000h, 030h, 030h, 030h, 060h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
518 00017921 00000000000000      <1>
519 00017928 00000C183030303030- <1>     db  000h, 000h, 00ch, 018h, 030h, 030h, 030h, 030h, 030h, 030h, 018h, 00ch, 000h, 000h, 000h, 000h
519 00017931 30180C00000000      <1>
520 00017938 000030180C0C0C0C0C- <1>     db  000h, 000h, 030h, 018h, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 018h, 030h, 000h, 000h, 000h, 000h
520 00017941 0C183000000000      <1>
521 00017948 0000000000663CFF3C- <1>     db  000h, 000h, 000h, 000h, 000h, 066h, 03ch, 0ffh, 03ch, 066h, 000h, 000h, 000h, 000h, 000h, 000h
521 00017951 66000000000000      <1>
522 00017958 000000000018187E18- <1>     db  000h, 000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h
522 00017961 18000000000000      <1>
523 00017968 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 018h, 030h, 000h, 000h, 000h
523 00017971 18181830000000      <1>
524 00017978 000000000000000FE00- <1>    db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
524 00017981 00000000000000      <1>
525 00017988 000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
525 00017991 00181800000000      <1>
526 00017998 0000000002060C1830- <1>     db  000h, 000h, 000h, 000h, 002h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 080h, 000h, 000h, 000h, 000h
526 000179A1 60C08000000000      <1>
527 000179A8 00003C66C3C3DBDBC3- <1>     db  000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h, 000h, 000h
527 000179B1 C3663C00000000      <1>
528 000179B8 000018387818181818- <1>     db  000h, 000h, 018h, 038h, 078h, 018h, 018h, 018h, 018h, 018h, 018h, 07eh, 000h, 000h, 000h, 000h
528 000179C1 18187E00000000      <1>
529 000179C8 00007CC6060C183060- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 00ch, 018h, 030h, 060h, 0c0h, 0c6h, 0feh, 000h, 000h, 000h, 000h
529 000179D1 C0C6FE00000000      <1>
530 000179D8 00007CC606063C0606- <1>     db  000h, 000h, 07ch, 0c6h, 006h, 006h, 03ch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
530 000179E1 06C67C00000000      <1>
531 000179E8 00000C1C3C6CCCFE0C- <1>     db  000h, 000h, 00ch, 01ch, 03ch, 06ch, 0cch, 0feh, 00ch, 00ch, 00ch, 01eh, 000h, 000h, 000h, 000h
531 000179F1 0C0C1E00000000      <1>
532 000179F8 0000FEC0C0C0FC0606- <1>     db  000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0fch, 006h, 006h, 006h, 0c6h, 07ch, 000h, 000h, 000h, 000h
532 00017A01 06C67C00000000      <1>
533 00017A08 00003860C0C0FCC6C6- <1>     db  000h, 000h, 038h, 060h, 0c0h, 0c0h, 0fch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
533 00017A11 C6C67C00000000      <1>
534 00017A18 0000FEC606060C1830- <1>     db  000h, 000h, 0feh, 0c6h, 006h, 006h, 00ch, 018h, 030h, 030h, 030h, 030h, 000h, 000h, 000h, 000h
534 00017A21 30303000000000      <1>
535 00017A28 00007CC6C6C67CC6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
535 00017A31 C6C67C00000000      <1>
536 00017A38 00007CC6C6C67E0606- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 07eh, 006h, 006h, 006h, 00ch, 078h, 000h, 000h, 000h, 000h
536 00017A41 060C7800000000      <1>
537 00017A48 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
537 00017A51 18180000000000      <1>
538 00017A58 000000001818000000- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 018h, 018h, 030h, 000h, 000h, 000h, 000h
538 00017A61 18183000000000      <1>
539 00017A68 000000060C18306030- <1>     db  000h, 000h, 000h, 006h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 006h, 000h, 000h, 000h, 000h
539 00017A71 180C0600000000      <1>
540 00017A78 00000000007E00007E- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 000h, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h
540 00017A81 00000000000000      <1>
541 00017A88 0000006030180C060C- <1>     db  000h, 000h, 000h, 060h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 060h, 000h, 000h, 000h, 000h
541 00017A91 18306000000000      <1>
542 00017A98 00007CC6C60C181818- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 00ch, 018h, 018h, 018h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
542 00017AA1 00181800000000      <1>
543 00017AA8 0000007CC6C6DEDEDE- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0deh, 0deh, 0deh, 0dch, 0c0h, 07ch, 000h, 000h, 000h, 000h
543 00017AB1 DCC07C00000000      <1>
544 00017AB8 000010386CC6C6FEC6- <1>     db  000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
544 00017AC1 C6C6C600000000      <1>
545 00017AC8 0000FC6666667C6666- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 066h, 066h, 066h, 066h, 0fch, 000h, 000h, 000h, 000h
545 00017AD1 6666FC00000000      <1>
546 00017AD8 00003C66C2C0C0C0C0- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 000h, 000h, 000h, 000h
546 00017AE1 C2663C00000000      <1>
547 00017AE8 0000F86C6666666666- <1>     db  000h, 000h, 0f8h, 06ch, 066h, 066h, 066h, 066h, 066h, 066h, 06ch, 0f8h, 000h, 000h, 000h, 000h
547 00017AF1 666CF800000000      <1>
548 00017AF8 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
548 00017B01 6266FE00000000      <1>
549 00017B08 0000FE666268786860- <1>     db  000h, 000h, 0feh, 066h, 062h, 068h, 078h, 068h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
549 00017B11 6060F000000000      <1>
550 00017B18 00003C66C2C0C0DEC6- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0deh, 0c6h, 0c6h, 066h, 03ah, 000h, 000h, 000h, 000h
550 00017B21 C6663A00000000      <1>
551 00017B28 0000C6C6C6C6FEC6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
551 00017B31 C6C6C600000000      <1>
552 00017B38 00003C181818181818- <1>     db  000h, 000h, 03ch, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
552 00017B41 18183C00000000      <1>
553 00017B48 00001E0C0C0C0C0CCC- <1>     db  000h, 000h, 01eh, 00ch, 00ch, 00ch, 00ch, 00ch, 0cch, 0cch, 0cch, 078h, 000h, 000h, 000h, 000h
553 00017B51 CCCC7800000000      <1>
554 00017B58 0000E666666C78786C- <1>     db  000h, 000h, 0e6h, 066h, 066h, 06ch, 078h, 078h, 06ch, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
554 00017B61 6666E600000000      <1>
555 00017B68 0000F06060606060C0- <1>     db  000h, 000h, 0f0h, 060h, 060h, 060h, 060h, 060h, 060h, 062h, 066h, 0feh, 000h, 000h, 000h, 000h
555 00017B71 6266FE00000000      <1>
556 00017B78 0000C3E7FFFFDBC3C3- <1>     db  000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
556 00017B81 C3C3C300000000      <1>
557 00017B88 0000C6E6F6FEDECEC6- <1>     db  000h, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
557 00017B91 C6C6C600000000      <1>
558 00017B98 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
558 00017BA1 C6C67C00000000      <1>
559 00017BA8 0000FC6666667C6060- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
559 00017BB1 6060F000000000      <1>
560 00017BB8 00007CC6C6C6C6C6C6- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0d6h, 0deh, 07ch, 00ch, 00eh, 000h, 000h
560 00017BC1 D6DE7C0C0E0000      <1>
561 00017BC8 0000FC6666667C6C66- <1>     db  000h, 000h, 0fch, 066h, 066h, 066h, 07ch, 06ch, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
561 00017BD1 6666E600000000      <1>
562 00017BD8 00007CC6C660380C06- <1>     db  000h, 000h, 07ch, 0c6h, 0c6h, 060h, 038h, 00ch, 006h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
562 00017BE1 C6C67C00000000      <1>
563 00017BE8 0000FFDB9918181818- <1>     db  000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
563 00017BF1 18183C00000000      <1>
564 00017BF8 0000C6C6C6C6C6C6C6- <1>     db  000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
564 00017C01 C6C67C00000000      <1>
565 00017C08 0000C3C3C3C3C3C3C3- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
565 00017C11 663C1800000000      <1>
566 00017C18 0000C3C3C3C3C3DBDB- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 000h
566 00017C21 FF666600000000      <1>
567 00017C28 0000C3C3663C18183C- <1>     db  000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 000h
567 00017C31 66C3C300000000      <1>
568 00017C38 0000C3C3C3663C1818- <1>     db  000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
568 00017C41 18183C00000000      <1>
569 00017C48 0000FFC3860C183060- <1>     db  000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h
569 00017C51 C1C3FF00000000      <1>
570 00017C58 00003C303030303030- <1>     db  000h, 000h, 03ch, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 030h, 03ch, 000h, 000h, 000h, 000h
570 00017C61 30303C00000000      <1>
571 00017C68 00000080C0E070381C- <1>     db  000h, 000h, 000h, 080h, 0c0h, 0e0h, 070h, 038h, 01ch, 00eh, 006h, 002h, 000h, 000h, 000h, 000h
571 00017C71 0E060200000000      <1>
```

```
572 00017C78 00003C0C0C0C0C0C0C- <1>     db  000h, 000h, 03ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 00ch, 03ch, 000h, 000h, 000h, 000h
572 00017C81 0C0C3C00000000      <1>
573 00017C88 10386CC60000000000- <1>     db  010h, 038h, 06ch, 0c6h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
573 00017C91 00000000000000      <1>
574 00017C98 00000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h
574 00017CA1 00000000FF0000      <1>
575 00017CA8 303018000000000000- <1>     db  030h, 030h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
575 00017CB1 00000000000000      <1>
576 00017CB8 0000000000780C7CCC- <1>     db  000h, 000h, 000h, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
576 00017CC1 CCCC7600000000      <1>
577 00017CC8 0000E06060786C6666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 078h, 06ch, 066h, 066h, 066h, 066h, 07ch, 000h, 000h, 000h, 000h
577 00017CD1 66667C00000000      <1>
578 00017CD8 00000000007CC6C0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c0h, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
578 00017CE1 C0C67C00000000      <1>
579 00017CE8 00001C0C0C3C6CCCCC- <1>     db  000h, 000h, 01ch, 00ch, 00ch, 03ch, 06ch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
579 00017CF1 CCCC7600000000      <1>
580 00017CF8 00000000007CC6FEC0- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
580 00017D01 C0C67C00000000      <1>
581 00017D08 0000386C6460F06060- <1>     db  000h, 000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
581 00017D11 6060F000000000      <1>
582 00017D18 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 0cch, 078h, 000h
582 00017D21 CCCC7C0CCC7800      <1>
583 00017D28 0000E060606C766666- <1>     db  000h, 000h, 0e0h, 060h, 060h, 06ch, 076h, 066h, 066h, 066h, 066h, 0e6h, 000h, 000h, 000h, 000h
583 00017D31 6666E600000000      <1>
584 00017D38 000018180038181818- <1>     db  000h, 000h, 018h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
584 00017D41 18183C00000000      <1>
585 00017D48 000006060006060606- <1>     db  000h, 000h, 006h, 006h, 000h, 00eh, 006h, 006h, 006h, 006h, 006h, 006h, 066h, 066h, 03ch, 000h
585 00017D51 06060666663C00      <1>
586 00017D58 0000E06060666C7878- <1>     db  000h, 000h, 0e0h, 060h, 060h, 066h, 06ch, 078h, 078h, 06ch, 066h, 0e6h, 000h, 000h, 000h, 000h
586 00017D61 6C66E600000000      <1>
587 00017D68 000038181818181818- <1>     db  000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
587 00017D71 18183C00000000      <1>
588 00017D78 0000000000E6FFDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h
588 00017D81 DBDBDB00000000      <1>
589 00017D88 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
589 00017D91 666666000000000     <1>
590 00017D98 00000000007CC6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
590 00017DA1 C6C67C00000000      <1>
591 00017DA8 0000000000DC666666- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0f0h, 000h
591 00017DB1 66667C6060F000      <1>
592 00017DB8 000000000076CCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0cch, 0cch, 0cch, 0cch, 0cch, 07ch, 00ch, 00ch, 01eh, 000h
592 00017DC1 CCCC7C0C0C1E00      <1>
593 00017DC8 0000000000DC766660- <1>     db  000h, 000h, 000h, 000h, 000h, 0dch, 076h, 066h, 060h, 060h, 060h, 0f0h, 000h, 000h, 000h, 000h
593 00017DD1 6060F000000000      <1>
594 00017DD8 00000000007CC66038- <1>     db  000h, 000h, 000h, 000h, 000h, 07ch, 0c6h, 060h, 038h, 00ch, 0c6h, 07ch, 000h, 000h, 000h, 000h
594 00017DE1 0CC67C00000000      <1>
595 00017DE8 0000103030FC303030- <1>     db  000h, 000h, 010h, 030h, 030h, 0fch, 030h, 030h, 030h, 030h, 036h, 01ch, 000h, 000h, 000h, 000h
595 00017DF1 30361C00000000      <1>
596 00017DF8 0000000000CCCCCCCC- <1>     db  000h, 000h, 000h, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
596 00017E01 CCCC7600000000      <1>
597 00017E08 0000000000C3C3C3C3- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h
597 00017E11 663C1800000000      <1>
598 00017E18 0000000000C3C3C3DB- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h
598 00017E21 DBFF6600000000      <1>
599 00017E28 0000000000C3663C18- <1>     db  000h, 000h, 000h, 000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h
599 00017E31 3C66C300000000      <1>
600 00017E38 0000000000C6C6C6C6- <1>     db  000h, 000h, 000h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 0f8h, 000h
600 00017E41 C6C67E060CF800      <1>
601 00017E48 0000000000FECC1830- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 0cch, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
601 00017E51 60C6FE00000000      <1>
602 00017E58 00000E181818701818- <1>     db  000h, 000h, 00eh, 018h, 018h, 018h, 070h, 018h, 018h, 018h, 018h, 00eh, 000h, 000h, 000h, 000h
602 00017E61 18180E00000000      <1>
603 00017E68 000018181818001818- <1>     db  000h, 000h, 018h, 018h, 018h, 018h, 000h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
603 00017E71 18181800000000      <1>
604 00017E78 0000701818180E1818- <1>     db  000h, 000h, 070h, 018h, 018h, 018h, 00eh, 018h, 018h, 018h, 018h, 070h, 000h, 000h, 000h, 000h
604 00017E81 18187000000000      <1>
605 00017E88 000076DC0000000000- <1>     db  000h, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
605 00017E91 00000000000000      <1>
606 00017E98 0000000010386CC6C6- <1>     db  000h, 000h, 000h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 0feh, 000h, 000h, 000h, 000h, 000h
606 00017EA1 C6FE0000000000      <1>
607 00017EA8 00003C66C2C0C0C0C2- <1>     db  000h, 000h, 03ch, 066h, 0c2h, 0c0h, 0c0h, 0c0h, 0c2h, 066h, 03ch, 00ch, 006h, 07ch, 000h, 000h
607 00017EB1 663C0C067C0000      <1>
608 00017EB8 0000CC0000CCCCCCCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
608 00017EC1 CCCC7600000000      <1>
609 00017EC8 000C1830007CC6FEC0- <1>     db  000h, 00ch, 018h, 030h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
609 00017ED1 C0C67C00000000      <1>
610 00017ED8 0010386C00780C7CCC- <1>     db  000h, 010h, 038h, 06ch, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
610 00017EE1 CCCC7600000000      <1>
611 00017EE8 0000CC0000780C7CCC- <1>     db  000h, 000h, 0cch, 000h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
611 00017EF1 CCCC7600000000      <1>
612 00017EF8 0060301800780C7CCC- <1>     db  000h, 060h, 030h, 018h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
612 00017F01 CCCC7600000000      <1>
613 00017F08 00386C3800780C7CCC- <1>     db  000h, 038h, 06ch, 038h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
613 00017F11 CCCC7600000000      <1>
614 00017F18 000000003C66606066- <1>     db  000h, 000h, 000h, 000h, 03ch, 066h, 060h, 060h, 066h, 03ch, 00ch, 006h, 03ch, 000h, 000h, 000h
614 00017F21 3C0C063C000000      <1>
615 00017F28 0010386C007CC6FEC0- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
615 00017F31 C0C67C00000000      <1>
616 00017F38 0000C600007CC6FEC0- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
616 00017F41 C0C67C00000000      <1>
617 00017F48 00603018007CC6FEC0- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0feh, 0c0h, 0c0h, 0c6h, 07ch, 000h, 000h, 000h, 000h
617 00017F51 C0C67C00000000      <1>
618 00017F58 0000660000038181818- <1>     db  000h, 000h, 066h, 000h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
618 00017F61 18183C00000000      <1>
619 00017F68 00183C660038181818- <1>     db  000h, 018h, 03ch, 066h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
619 00017F71 18183C00000000      <1>
620 00017F78 0060301800038181818- <1>     db  000h, 060h, 030h, 018h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
620 00017F81 18183C00000000      <1>
621 00017F88 00C60010386CC6C6FE- <1>     db  000h, 0c6h, 000h, 010h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
621 00017F91 C6C6C600000000      <1>
622 00017F98 386C3800386CC6C6FE- <1>     db  038h, 06ch, 038h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
622 00017FA1 C6C6C600000000      <1>
623 00017FA8 18306000FE66607C60- <1>     db  018h, 030h, 060h, 000h, 0feh, 066h, 060h, 07ch, 060h, 060h, 066h, 0feh, 000h, 000h, 000h, 000h
623 00017FB1 6066FE00000000      <1>
624 00017FB8 00000000006E3B1B7E- <1>     db  000h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h
624 00017FC1 D8DC7700000000      <1>
625 00017FC8 00003E6CCCCCFECCCC- <1>     db  000h, 000h, 03eh, 06ch, 0cch, 0cch, 0feh, 0cch, 0cch, 0cch, 0cch, 0ceh, 000h, 000h, 000h, 000h
625 00017FD1 CCCCCE00000000      <1>
626 00017FD8 0010386C007CC6C6C6- <1>     db  000h, 010h, 038h, 06ch, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
626 00017FE1 C6C67C00000000      <1>
627 00017FE8 0000C600007CC6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
```

```
627 00017FF1 C6C67C00000000      <1>
628 00017FF8 00603018007CC6C6C6- <1>     db  000h, 060h, 030h, 018h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
628 00018001 C6C67C00000000      <1>
629 00018008 003078CC00CCCCCCCC- <1>     db  000h, 030h, 078h, 0cch, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
629 00018011 CCCC7600000000      <1>
630 00018018 0060301800CCCCCCCC- <1>     db  000h, 060h, 030h, 018h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
630 00018021 CCCC7600000000      <1>
631 00018028 0000C60000C6C6C6C6- <1>     db  000h, 000h, 0c6h, 000h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07eh, 006h, 00ch, 078h, 000h
631 00018031 C6C67E060C7800      <1>
632 00018038 00C6007CC6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
632 00018041 C6C67C00000000      <1>
633 00018048 00C600C6C6C6C6C6C6- <1>     db  000h, 0c6h, 000h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
633 00018051 C6C67C00000000      <1>
634 00018058 0018187EC3C0C0C0C3- <1>     db  000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h
634 00018061 7E181800000000      <1>
635 00018068 00386C6460F0606060- <1>     db  000h, 038h, 06ch, 064h, 060h, 0f0h, 060h, 060h, 060h, 060h, 0e6h, 0fch, 000h, 000h, 000h, 000h
635 00018071 60E6FC00000000      <1>
636 00018078 0000C3663C18FF18FF- <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
636 00018081 18181800000000      <1>
637 00018088 00FC66667C62666F66- <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h, 000h
637 00018091 6666F300000000      <1>
638 00018098 000E1B1818187E1818- <1>     db  000h, 00eh, 01bh, 018h, 018h, 018h, 07eh, 018h, 018h, 018h, 018h, 018h, 0d8h, 070h, 000h, 000h
638 000180A1 181818D8700000      <1>
639 000180A8 0018306000780C7CCC- <1>     db  000h, 018h, 030h, 060h, 000h, 078h, 00ch, 07ch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
639 000180B1 CCCC7600000000      <1>
640 000180B8 000C18300038181818- <1>     db  000h, 00ch, 018h, 030h, 000h, 038h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 000h
640 000180C1 18183C00000000      <1>
641 000180C8 00183060007CC6C6C6- <1>     db  000h, 018h, 030h, 060h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
641 000180D1 C6C67C00000000      <1>
642 000180D8 0018306000CCCCCCCC- <1>     db  000h, 018h, 030h, 060h, 000h, 0cch, 0cch, 0cch, 0cch, 0cch, 0cch, 076h, 000h, 000h, 000h, 000h
642 000180E1 CCCC7600000000      <1>
643 000180E8 000076DC00DC666666- <1>     db  000h, 000h, 076h, 0dch, 000h, 0dch, 066h, 066h, 066h, 066h, 066h, 066h, 000h, 000h, 000h, 000h
643 000180F1 66666600000000      <1>
644 000180F8 76DC00C6E6F6FEDECE- <1>     db  076h, 0dch, 000h, 0c6h, 0e6h, 0f6h, 0feh, 0deh, 0ceh, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
644 00018101 C6C6C600000000      <1>
645 00018108 003C6C6C3E007E0000- <1>     db  000h, 03ch, 06ch, 06ch, 03eh, 000h, 07eh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
645 00018111 00000000000000      <1>
646 00018118 00386C6C38007C0000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 07ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
646 00018121 00000000000000      <1>
647 00018128 0000303000303060C0- <1>     db  000h, 000h, 030h, 030h, 000h, 030h, 030h, 060h, 0c0h, 0c6h, 0c6h, 07ch, 000h, 000h, 000h, 000h
647 00018131 C6C67C00000000      <1>
648 00018138 000000000000FEC0C0- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h, 000h
648 00018141 C0C000000000000     <1>
649 00018148 000000000000FE0606- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 0feh, 006h, 006h, 006h, 006h, 000h, 000h, 000h, 000h, 000h
649 00018151 060600000000000     <1>
650 00018158 00C0C0C2C6CC183060- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh, 000h, 000h
650 00018161 CE9B060C1F0000      <1>
651 00018168 00C0C0C2C6CC183066- <1>     db  000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h, 006h, 000h, 000h
651 00018171 CE963E06060000      <1>
652 00018178 000018180018183C- <1>     db  000h, 000h, 018h, 018h, 000h, 018h, 018h, 018h, 03ch, 03ch, 03ch, 018h, 000h, 000h, 000h, 000h
652 00018181 3C3C1800000000      <1>
653 00018188 0000000000366CD86C- <1>     db  000h, 000h, 000h, 000h, 000h, 036h, 06ch, 0d8h, 06ch, 036h, 000h, 000h, 000h, 000h, 000h, 000h
653 00018191 36000000000000      <1>
654 00018198 0000000000D86C366C- <1>     db  000h, 000h, 000h, 000h, 000h, 0d8h, 06ch, 036h, 06ch, 0d8h, 000h, 000h, 000h, 000h, 000h, 000h
654 000181A1 D8000000000000      <1>
655 000181A8 114411441144114411- <1>     db  011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h, 011h, 044h
655 000181B1 44114411441144      <1>
656 000181B8 55AA55AA55AA55AA55- <1>     db  055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah, 055h, 0aah
656 000181C1 AA55AA55AA55AA      <1>
657 000181C8 DD77DD77DD77DD77DD- <1>     db  0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h, 0ddh, 077h
657 000181D1 77DD77DD77DD77      <1>
658 000181D8 181818181818181818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
658 000181E1 18181818181818      <1>
659 000181E8 18181818181818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
659 000181F1 18181818181818      <1>
660 000181F8 1818181818F818F818- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
660 00018201 18181818181818      <1>
661 00018208 36363636363636F636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
661 00018211 36363636363636      <1>
662 00018218 000000000000000FE36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0feh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
662 00018221 36363636363636      <1>
663 00018228 0000000000F818F818- <1>     db  000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
663 00018231 18181818181818      <1>
664 00018238 3636363636F606F636- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
664 00018241 36363636363636      <1>
665 00018248 3636363636363636- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
665 00018251 36363636363636      <1>
666 00018258 0000000000FE06F636- <1>     db  000h, 000h, 000h, 000h, 000h, 0feh, 006h, 0f6h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
666 00018261 36363636363636      <1>
667 00018268 3636363636F606FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f6h, 006h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
667 00018271 00000000000000      <1>
668 00018278 36363636363636FE00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0feh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
668 00018281 00000000000000      <1>
669 00018288 1818181818F818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 0f8h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
669 00018291 00000000000000      <1>
670 00018298 000000000000000F818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0f8h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
670 000182A1 18181818181818      <1>
671 000182A8 181818181818181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
671 000182B1 00000000000000      <1>
672 000182B8 181818181818FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
672 000182C1 00000000000000      <1>
673 000182C8 000000000000FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
673 000182D1 18181818181818      <1>
674 000182D8 181818181818181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
674 000182E1 18181818181818      <1>
675 000182E8 000000000000FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
675 000182F1 00000000000000      <1>
676 000182F8 181818181818FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
676 00018301 18181818181818      <1>
677 00018308 18181818181F181F18- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
677 00018311 18181818181818      <1>
678 00018318 3636363636363736- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
678 00018321 36363636363636      <1>
679 00018328 3636363636373F00- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
679 00018331 00000000000000      <1>
680 00018338 00000000003F373736- <1>     db  000h, 000h, 000h, 000h, 000h, 03fh, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
680 00018341 36363636363636      <1>
681 00018348 3636363636F700FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
681 00018351 00000000000000      <1>
682 00018358 0000000000FF00F736- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
682 00018361 36363636363636      <1>
```

```
683 00018368 363636363637303736- <1>     db  036h, 036h, 036h, 036h, 036h, 037h, 030h, 037h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
683 00018371 36363636363636      <1>
684 00018378 00000000000FF00FF00- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
684 00018381 00000000000000      <1>
685 00018388 3636363636F700F736- <1>     db  036h, 036h, 036h, 036h, 036h, 0f7h, 000h, 0f7h, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
685 00018391 36363636363636      <1>
686 00018398 1818181818FF00FF00- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
686 000183A1 00000000000000      <1>
687 000183A8 363636363636FF00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
687 000183B1 00000000000000      <1>
688 000183B8 0000000000FF00FF18- <1>     db  000h, 000h, 000h, 000h, 000h, 0ffh, 000h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
688 000183C1 18181818181818      <1>
689 000183C8 00000000000000FF36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
689 000183D1 36363636363636      <1>
690 000183D8 3636363636363F00- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 03fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
690 000183E1 00000000000000      <1>
691 000183E8 18181818181F181F00- <1>     db  018h, 018h, 018h, 018h, 018h, 01fh, 018h, 01fh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
691 000183F1 00000000000000      <1>
692 000183F8 00000000001F181F18- <1>     db  000h, 000h, 000h, 000h, 000h, 01fh, 018h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
692 00018401 18181818181818      <1>
693 00018408 000000000000003F36- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 03fh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
693 00018411 36363636363636      <1>
694 00018418 363636363636FF36- <1>     db  036h, 036h, 036h, 036h, 036h, 036h, 036h, 0ffh, 036h, 036h, 036h, 036h, 036h, 036h, 036h, 036h
694 00018421 36363636363636      <1>
695 00018428 1818181818FF18FF18- <1>     db  018h, 018h, 018h, 018h, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
695 00018431 18181818181818      <1>
696 00018438 181818181818F800- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
696 00018441 00000000000000      <1>
697 00018448 000000000000001F18- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 01fh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
697 00018451 18181818181818      <1>
698 00018458 FFFFFFFFFFFFFFFFFF- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
698 00018461 FFFFFFFFFFFFFF      <1>
699 00018468 000000000000000FFFF- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh
699 00018471 FFFFFFFFFFFFFF      <1>
700 00018478 F0F0F0F0F0F0F0F0F0- <1>     db  0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h, 0f0h
700 00018481 F0F0F0F0F0F0F0      <1>
701 00018488 0F0F0F0F0F0F0F0F0F- <1>     db  00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh, 00fh
701 00018491 0F0F0F0F0F0F0F      <1>
702 00018498 FFFFFFFFFFFFFF0000- <1>     db  0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
702 000184A1 00000000000000      <1>
703 000184A8 0000000000076DCD8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 0d8h, 0d8h, 0d8h, 0dch, 076h, 000h, 000h, 000h, 000h
703 000184B1 D8DC7600000000      <1>
704 000184B8 000078CCCCCCD8CCC6- <1>     db  000h, 000h, 078h, 0cch, 0cch, 0cch, 0d8h, 0cch, 0c6h, 0c6h, 0c6h, 0cch, 000h, 000h, 000h, 000h
704 000184C1 C6C6CC00000000      <1>
705 000184C8 0000FEC6C6C0C0C0C0- <1>     db  000h, 000h, 0feh, 0c6h, 0c6h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 0c0h, 000h, 000h, 000h, 000h
705 000184D1 C0C0C000000000      <1>
706 000184D8 00000000FE6C6C6C6C- <1>     db  000h, 000h, 000h, 000h, 0feh, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h
706 000184E1 6C6C6C00000000      <1>
707 000184E8 000000FEC660301830- <1>     db  000h, 000h, 000h, 0feh, 0c6h, 060h, 030h, 018h, 030h, 060h, 0c6h, 0feh, 000h, 000h, 000h, 000h
707 000184F1 60C6FE00000000      <1>
708 000184F8 00000000007ED8D8D8- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0d8h, 0d8h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
708 00018501 D8D87000000000      <1>
709 00018508 0000000066666666666- <1>     db  000h, 000h, 000h, 000h, 066h, 066h, 066h, 066h, 066h, 07ch, 060h, 060h, 0c0h, 000h, 000h, 000h
709 00018511 7C6060C0000000      <1>
710 00018518 0000000076DC181818- <1>     db  000h, 000h, 000h, 000h, 076h, 0dch, 018h, 018h, 018h, 018h, 018h, 018h, 000h, 000h, 000h, 000h
710 00018521 18181800000000      <1>
711 00018528 0000007E183C666666- <1>     db  000h, 000h, 000h, 07eh, 018h, 03ch, 066h, 066h, 066h, 03ch, 018h, 07eh, 000h, 000h, 000h, 000h
711 00018531 3C187E00000000      <1>
712 00018538 000000386CC6C6FEC6- <1>     db  000h, 000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0feh, 0c6h, 0c6h, 06ch, 038h, 000h, 000h, 000h, 000h
712 00018541 C66C3800000000      <1>
713 00018548 0000386CC6C6C66C6C- <1>     db  000h, 000h, 038h, 06ch, 0c6h, 0c6h, 0c6h, 06ch, 06ch, 06ch, 06ch, 0eeh, 000h, 000h, 000h, 000h
713 00018551 6C6CEE00000000      <1>
714 00018558 00001E30180C3E6666- <1>     db  000h, 000h, 01eh, 030h, 018h, 00ch, 03eh, 066h, 066h, 066h, 066h, 03ch, 000h, 000h, 000h, 000h
714 00018561 66663C00000000      <1>
715 00018568 00000000007EDBDBDB- <1>     db  000h, 000h, 000h, 000h, 000h, 07eh, 0dbh, 0dbh, 0dbh, 07eh, 000h, 000h, 000h, 000h, 000h, 000h
715 00018571 7E000000000000      <1>
716 00018578 00000003067EDBDBF3- <1>     db  000h, 000h, 000h, 003h, 006h, 07eh, 0dbh, 0dbh, 0f3h, 07eh, 060h, 0c0h, 000h, 000h, 000h, 000h
716 00018581 7E60C000000000      <1>
717 00018588 00001C3060607C6060- <1>     db  000h, 000h, 01ch, 030h, 060h, 060h, 07ch, 060h, 060h, 060h, 030h, 01ch, 000h, 000h, 000h, 000h
717 00018591 60301C00000000      <1>
718 00018598 0000007CC6C6C6C6C6- <1>     db  000h, 000h, 000h, 07ch, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 0c6h, 000h, 000h, 000h, 000h
718 000185A1 C6C6C600000000      <1>
719 000185A8 00000000FE0000FE00- <1>     db  000h, 000h, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 0feh, 000h, 000h, 000h, 000h, 000h
719 000185B1 00FE0000000000      <1>
720 000185B8 0000000018187E1818- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 07eh, 018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h
720 000185C1 0000FF00000000      <1>
721 000185C8 00000030180C060C18- <1>     db  000h, 000h, 000h, 030h, 018h, 00ch, 006h, 00ch, 018h, 030h, 000h, 07eh, 000h, 000h, 000h, 000h
721 000185D1 30007E00000000      <1>
722 000185D8 0000000C1830603018- <1>     db  000h, 000h, 000h, 00ch, 018h, 030h, 060h, 030h, 018h, 00ch, 000h, 07eh, 000h, 000h, 000h, 000h
722 000185E1 0C007E00000000      <1>
723 000185E8 00000E1B1B18181818- <1>     db  000h, 000h, 00eh, 01bh, 01bh, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h
723 000185F1 18181818181818      <1>
724 000185F8 1818181818181818D8- <1>     db  018h, 018h, 018h, 018h, 018h, 018h, 018h, 018h, 0d8h, 0d8h, 0d8h, 070h, 000h, 000h, 000h, 000h
724 00018601 D8D87000000000      <1>
725 00018608 000000001818007E00- <1>     db  000h, 000h, 000h, 000h, 018h, 018h, 000h, 07eh, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h
725 00018611 18180000000000      <1>
726 00018618 0000000076DC0076- <1>     db  000h, 000h, 000h, 000h, 000h, 076h, 0dch, 000h, 076h, 0dch, 000h, 000h, 000h, 000h, 000h, 000h
726 00018621 DC000000000000      <1>
727 00018628 00386C6C3800000000- <1>     db  000h, 038h, 06ch, 06ch, 038h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
727 00018631 00000000000000      <1>
728 00018638 000000000000001818- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
728 00018641 00000000000000      <1>
729 00018648 000000000000000018- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 018h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
729 00018651 00000000000000      <1>
730 00018658 000F0C0C0C0C0CEC6C- <1>     db  000h, 00fh, 00ch, 00ch, 00ch, 00ch, 00ch, 0ech, 06ch, 06ch, 03ch, 01ch, 000h, 000h, 000h, 000h
730 00018661 6C3C1C00000000      <1>
731 00018668 00D86C6C6C6C6C0000- <1>     db  000h, 0d8h, 06ch, 06ch, 06ch, 06ch, 06ch, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
731 00018671 00000000000000      <1>
732 00018678 0070D83060C8F80000- <1>     db  000h, 070h, 0d8h, 030h, 060h, 0c8h, 0f8h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
732 00018681 00000000000000      <1>
733 00018688 000000007C7C7C7C7C- <1>     db  000h, 000h, 000h, 000h, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 07ch, 000h, 000h, 000h, 000h, 000h
733 00018691 7C7C0000000000      <1>
734 00018698 0000000000000000000- <1>     db  000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h
734 000186A1 00000000000000      <1>
735                              <1>
736                              <1> ; 01/01/2021 (TRDOS 386 v2.0.3)
737                              <1>
738                              <1> %if 0
739                              <1>
740                              <1> vgafont14alt:
741                              <1>     db  01dh, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h, 022h
```

```
                        <1>     db  000h, 063h, 063h, 063h, 022h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 02bh, 000h
                        <1>     db  000h, 000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h, 02dh, 000h, 000h
                        <1>     db  000h, 000h, 000h, 000h, 0ffh, 000h, 000h, 000h, 000h, 000h, 000h, 000h, 04dh, 000h, 000h, 0c3h
                        <1>     db  0e7h, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh
                        <1>     db  099h, 018h, 018h, 018h, 018h, 018h, 03ch, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h
                        <1>     db  0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h
                        <1>     db  0dbh, 0dbh, 0ffh, 066h, 066h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h
                        <1>     db  03ch, 066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
                        <1>     db  018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h, 030h, 061h
                        <1>     db  0c3h, 0ffh, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h, 0ffh, 0dbh, 0dbh, 0dbh
                        <1>     db  0dbh, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
                        <1>     db  000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h, 000h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h
                        <1>     db  000h, 000h, 091h, 000h, 000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h
                        <1>     db  000h, 09bh, 000h, 018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h
                        <1>     db  09dh, 000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 000h, 000h, 000h, 09eh
                        <1>     db  000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 0f3h, 000h, 000h, 000h, 0f1h, 000h
                        <1>     db  000h, 018h, 018h, 018h, 0ffh, 018h, 018h, 018h, 000h, 0ffh, 000h, 000h, 000h, 0f6h, 000h, 000h
                        <1>     db  018h, 018h, 000h, 000h, 0ffh, 000h, 000h, 018h, 018h, 000h, 000h, 000h, 000h
                        <1> vgafont16alt:
                        <1>     db  01dh, 000h, 000h, 000h, 000h, 000h, 024h, 066h, 0ffh, 066h, 024h, 000h, 000h, 000h, 000h, 000h
                        <1>     db  000h, 030h, 000h, 000h, 03ch, 066h, 0c3h, 0c3h, 0dbh, 0dbh, 0c3h, 0c3h, 066h, 03ch, 000h, 000h
                        <1>     db  000h, 000h, 04dh, 000h, 000h, 0c3h, 0e7h, 0ffh, 0ffh, 0dbh, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 000h
                        <1>     db  000h, 000h, 000h, 054h, 000h, 000h, 0ffh, 0dbh, 099h, 018h, 018h, 018h, 018h, 018h, 018h, 03ch
                        <1>     db  000h, 000h, 000h, 000h, 056h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch
                        <1>     db  018h, 000h, 000h, 000h, 000h, 057h, 000h, 000h, 0c3h, 0c3h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh
                        <1>     db  066h, 066h, 000h, 000h, 000h, 000h, 058h, 000h, 000h, 0c3h, 0c3h, 066h, 03ch, 018h, 018h, 03ch
                        <1>     db  066h, 0c3h, 0c3h, 000h, 000h, 000h, 059h, 000h, 000h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h
                        <1>     db  018h, 018h, 018h, 03ch, 000h, 000h, 000h, 05ah, 000h, 000h, 0ffh, 0c3h, 086h, 00ch, 018h
                        <1>     db  030h, 060h, 0c1h, 0c3h, 0ffh, 000h, 000h, 000h, 000h, 06dh, 000h, 000h, 000h, 000h, 000h, 0e6h
                        <1>     db  0ffh, 0dbh, 0dbh, 0dbh, 0dbh, 0dbh, 000h, 000h, 000h, 000h, 076h, 000h, 000h, 000h, 000h, 000h
                        <1>     db  0c3h, 0c3h, 0c3h, 0c3h, 066h, 03ch, 018h, 000h, 000h, 000h, 000h, 077h, 000h, 000h, 000h, 000h
                        <1>     db  000h, 0c3h, 0c3h, 0c3h, 0dbh, 0dbh, 0ffh, 066h, 000h, 000h, 000h, 000h, 078h, 000h, 000h, 000h
                        <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 03ch, 066h, 0c3h, 000h, 000h, 000h, 000h, 091h, 000h, 000h
                        <1>     db  000h, 000h, 000h, 06eh, 03bh, 01bh, 07eh, 0d8h, 0dch, 077h, 000h, 000h, 000h, 000h, 09bh, 000h
                        <1>     db  018h, 018h, 07eh, 0c3h, 0c0h, 0c0h, 0c0h, 0c3h, 07eh, 018h, 018h, 000h, 000h, 000h, 000h, 09dh
                        <1>     db  000h, 000h, 0c3h, 066h, 03ch, 018h, 0ffh, 018h, 0ffh, 018h, 018h, 018h, 000h, 000h, 000h, 000h
                        <1>     db  09eh, 000h, 0fch, 066h, 066h, 07ch, 062h, 066h, 06fh, 066h, 066h, 066h, 0f3h, 000h, 000h, 000h
                        <1>     db  000h, 0abh, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 060h, 0ceh, 09bh, 006h, 00ch, 01fh
                        <1>     db  000h, 000h, 0ach, 000h, 0c0h, 0c0h, 0c2h, 0c6h, 0cch, 018h, 030h, 066h, 0ceh, 096h, 03eh, 006h
                        <1>     db  006h, 000h, 000h, 000h
                        <1>
                        <1> %endif
                                
                                    ; 20/11/2020
                                    vbe2_bochs_vbios:
                                    ;   db "BOCHS/QEMU"
000186A8 424F4348532F51454D-            db "BOCHS/QEMU/VIRTUALBOX"  ; 26/11/2020
000186B1 552F5649525455414C-
000186BA 424F58
                                    ;vbe_vnumber equ vbe2_bochs_vbios + 28 ; "3" or "2"
                                    ; 26/11/2020
                                    vbe_vnumber equ vbe2_bochs_vbios + 30 ; "3" or "2"
                                    ; 15/11/2020
                                    vesa_vbe3_bios_msg:
                                        ;db " VESA VBE version 3 Video BIOS ..."
000186BD 205645534120564245-            db " VESA VBE3 Video BIOS ..." ; 26/11/2020
000186C6 3320566964656F2042-
000186CF 494F53202E2E2E
000186D6 0D0A0D0A00                     db 0Dh, 0Ah, 0Dh, 0Ah, 0

                                    ; 28/02/2021
000186DB 18                         truecolor: db 24

                                    align 2

                                    ; EPOCH Variables
                                    ; 13/04/2015 - Retro UNIX 386 v1 Beginning
                                    ; 09/04/2013 epoch variables
                                    ; Retro UNIX 8086 v1 Prototype: UNIXCOPY.ASM, 10/03/2013
                                    ;
000186DC B207                       year:       dw 1970
000186DE 0100                       month:      dw 1
000186E0 0100                       day:  dw 1
000186E2 0000                       hour:       dw 0
000186E4 0000                       minute: dw 0
000186E6 0000                       second: dw 0

                                    DMonth:
000186E8 0000                           dw 0
000186EA 1F00                           dw 31
000186EC 3B00                           dw 59
000186EE 5A00                           dw 90
000186F0 7800                           dw 120
000186F2 9700                           dw 151
000186F4 B500                           dw 181
000186F6 D400                           dw 212
000186F8 F300                           dw 243
000186FA 1101                           dw 273
000186FC 3001                           dw 304
000186FE 4E01                           dw 334

                                    ; 15/11/2020
00018700 00                         db    0
                                    kernel_version_msg: ; 20/11/2020
00018701 5452444F5320283338-        db    "TRDOS (386) Kernel v2.0.3 by Erdogan Tan"
0001870A 3629204B65726E656C-
00018713 2076322E302E332062-
0001871C 79204572646F67616E-
00018725 2054616E
00018729 00                         db    0

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

                                    bss_start:

                                    ABSOLUTE bss_start
```

```
3494
3495 0001872A <res 00000006>          alignb 8 ; 25/12/2016
3496
3497                                       ; 15/04/2016
3498                                       ; TRDOS 386 (TRDOS v2.0)
3499                                       ;      80 interrupts
3500                                       ; 11/03/2015
3501                                       ; Interrupt Descriptor Table (20/08/2014)
3502                                  idt:
3503                                       ;resb  64*8 ; INT 0 to INT 3Fh
3504                                       ; 15/04/2016
3505 00018730 <res 00000280>              resb   80*8 ; INT 0 to INT 4Fh
3506
3507                                  idt_end:
3508
3509                                  ;alignb 4
3510
3511                                  task_state_segment:
3512                                       ; 24/03/2015
3513 000189B0 <res 00000002>          tss.link:   resw 1
3514 000189B2 <res 00000002>                      resw 1
3515                                  ; tss offset 4
3516 000189B4 <res 00000004>          tss.esp0:   resd 1
3517 000189B8 <res 00000002>          tss.ss0:    resw 1
3518 000189BA <res 00000002>                      resw 1
3519 000189BC <res 00000004>          tss.esp1:   resd 1
3520 000189C0 <res 00000002>          tss.ss1:    resw 1
3521 000189C2 <res 00000002>                      resw 1
3522 000189C4 <res 00000004>          tss.esp2:   resd 1
3523 000189C8 <res 00000002>          tss.ss2:    resw 1
3524 000189CA <res 00000002>                      resw 1
3525                                  ; tss offset 28
3526 000189CC <res 00000004>          tss.CR3:    resd 1
3527 000189D0 <res 00000004>          tss.eip:    resd 1
3528 000189D4 <res 00000004>          tss.eflags: resd 1
3529                                  ; tss offset 40
3530 000189D8 <res 00000004>          tss.eax:    resd 1
3531 000189DC <res 00000004>          tss.ecx:    resd 1
3532 000189E0 <res 00000004>          tss.edx:    resd 1
3533 000189E4 <res 00000004>          tss.ebx:    resd 1
3534 000189E8 <res 00000004>          tss.esp:    resd 1
3535 000189EC <res 00000004>          tss.ebp:    resd 1
3536 000189F0 <res 00000004>          tss.esi:    resd 1
3537 000189F4 <res 00000004>          tss.edi:    resd 1
3538                                  ; tss offset 72
3539 000189F8 <res 00000002>          tss.ES:     resw 1
3540 000189FA <res 00000002>                    resw 1
3541 000189FC <res 00000002>          tss.CS:           resw 1
3542 000189FE <res 00000002>                    resw 1
3543 00018A00 <res 00000002>          tss.SS:           resw 1
3544 00018A02 <res 00000002>                    resw 1
3545 00018A04 <res 00000002>          tss.DS:           resw 1
3546 00018A06 <res 00000002>                    resw 1
3547 00018A08 <res 00000002>          tss.FS:           resw 1
3548 00018A0A <res 00000002>                    resw 1
3549 00018A0C <res 00000002>          tss.GS:           resw 1
3550 00018A0E <res 00000002>                    resw 1
3551 00018A10 <res 00000002>          tss.LDTR:   resw 1
3552 00018A12 <res 00000002>                    resw 1
3553                                  ; tss offset 100
3554 00018A14 <res 00000002>                    resw 1
3555 00018A16 <res 00000002>          tss.IOPB:   resw 1
3556                                  ; tss offset 104
3557                                  tss_end:
3558
3559 00018A18 <res 00000004>          k_page_dir:  resd 1 ; Kernel's (System) Page Directory address
3560                                                      ; (Physical address = Virtual address)
3561 00018A1C <res 00000004>          memory_size: resd 1 ; memory size in pages
3562 00018A20 <res 00000004>          free_pages:  resd 1 ; number of free pages
3563 00018A24 <res 00000004>          next_page:   resd 1 ; offset value in M.A.T. for
3564                                                      ; first free page search
3565 00018A28 <res 00000004>          last_page:   resd 1 ; offset value in M.A.T. which
3566                                                      ; next free page search will be
3567                                                      ; stopped after it. (end of M.A.T.)
3568 00018A2C <res 00000004>          first_page:  resd 1 ; offset value in M.A.T. which
3569                                                      ; first free page search
3570                                                      ; will be started on it. (for user)
3571 00018A30 <res 00000004>          mat_size:    resd 1 ; Memory Allocation Table size in pages
3572
3573                                  ; 20/11/2020
3574                                  ;vbe2bios:    resw 1 ; VBE2 video bios ID (bochs/qemu)
3575                                  ;                    ; (0B0C4h or 0B0C5h for bochs/plex86 vgabios)
3576
3577                                  ; 02/09/2014 (Retro UNIX 386 v1)
3578                                  ; 04/12/2013 (Retro UNIX 8086 v1)
3579 00018A34 <res 00000002>          CRT_START:   resw 1         ; starting address in regen buffer
3580                                                  ; NOTE: active page only
3581 00018A36 <res 00000010>          CURSOR_POSN: resw 8 ; cursor positions for video pages
3582                                  ACTIVE_PAGE:
3583 00018A46 <res 00000001>          ptty:             resb 1 ; current tty
3584                                  ; 01/07/2015 - 29/01/2016
3585 00018A47 <res 00000001>          ccolor:           resb 1 ; current color attribute
3586                                  ; 26/10/2015
3587                                  ; 07/09/2014
3588 00018A48 <res 00000014>          ttychr:      resw ntty+2 ; Character buffer (multiscreen)
3589
3590                                  ; 18/05/2015 (03/06/2013 - Retro UNIX 8086 v1 feature only!)
3591 00018A5C <res 00000004>          p_time:      resd 1     ; present time (for systime & sysmdate)
3592
3593                                  ; 18/05/2015 (16/08/2013 - Retro UNIX 8086 v1 feature only !)
3594                                  ; (open mode locks for pseudo TTYs)
3595                                  ; [ major tty locks (return error in any conflicts) ]
3596 00018A60 <res 00000014>          ttyl:        resw ntty+2 ; opening locks for TTYs.
3597
3598                                  ; 15/04/2015 (Retro UNIX 386 v1)
```

```
                                ; 22/09/2013 (Retro UNIX 8086 v1)
00018A74 <res 0000000A>         wlist:      resb ntty+2 ; wait channel list (0 to 9 for TTYs)
                                ; 15/04/2015 (Retro UNIX 386 v1)
                                ;; 12/07/2014 -> sp_init set comm. parameters as 0E3h
                                ;; 0 means serial port is not available
                                ;;comprm: ; 25/06/2014
00018A7E <res 00000001>         com1p:      resb 1  ;;0E3h
00018A7F <res 00000001>         com2p:      resb 1  ;;0E3h

                                ; 17/11/2015
                                ; request for response (from the terminal)
00018A80 <res 00000002>         req_resp:   resw 1
                                ; 07/11/2015
00018A82 <res 00000001>         ccomport:   resb 1 ; current COM (serial) port
                                           ; (0= COM1, 1= COM2)
                                ; 09/11/2015
00018A83 <res 00000001>         comqr:          resb 1 ; 'query or response' sign (u9.s, 'sndc')
                                ; 07/11/2015
00018A84 <res 00000002>         rchar:          resw 1 ; last received char for COM 1 and COM 2
00018A86 <res 00000002>         schar:          resw 1 ; last sent char for COM 1 and COM 2

                                ; 22/08/2014 (RTC)
                                ; (Packed BCD)
00018A88 <res 00000001>         time_seconds: resb 1
00018A89 <res 00000001>         time_minutes: resb 1
00018A8A <res 00000001>         time_hours:   resb 1
00018A8B <res 00000001>         date_wday:    resb 1
00018A8C <res 00000001>         date_day:     resb 1
00018A8D <res 00000001>         date_month:   resb 1
00018A8E <res 00000001>         date_year:    resb 1
00018A8F <res 00000001>         date_century: resb 1

                                ; 24/01/2016
00018A90 <res 00000004>         RTC_LH:           resd 1
00018A94 <res 00000001>         RTC_WAIT_FLAG: resb 1
00018A95 <res 00000001>         USER_FLAG:     resb 1
                                ; 19/05/2016
                                ;RTC_second:
00018A96 <res 00000001>         RTC_2Hz:       resb 1 ;  from 2Hz interrupt to 1Hz timer event function

                                %include 'diskbss.s'     ; UNINITIALIZED DISK (BIOS) DATA
                              <1> ; ****************************************************************************
                              <1> ; TRDOS386.ASM (TRDOS 386 Kernel) - v2.0.0 - diskbss.s
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Last Update: 24/01/2016
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Beginning: 24/01/2016
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Assembler: NASM version 2.11 (trdos386.s)
                              <1> ; ----------------------------------------------------------------------------
                              <1> ; Turkish Rational DOS
                              <1> ; Operating System Project v2.0 by ERDOGAN TAN (Beginning: 04/01/2016)
                              <1> ;
                              <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                              <1> ; diskbss.inc (10/07/2015)
                              <1> ;
                              <1> ; Derived from 'IBM PC-XT-286' BIOS source code (1986)
                              <1> ; ****************************************************************************
                              <1>
                              <1> ; Retro UNIX 386 v1 Kernel - DISKBSS.INC
                              <1> ; Last Modification: 10/07/2015
                              <1> ;     (Unnitialized Disk Parameters Data section for 'DISKIO.INC')
                              <1>
00018A97 <res 00000001>       <1> alignb 2
                              <1>
                              <1> ;----------------------------------------
                              <1> ;     TIMER DATA AREA           :
                              <1> ;----------------------------------------
                              <1>
                              <1> TIMER_LH:    ; 16/02/205
00018A98 <res 00000002>       <1> TIMER_LOW:      resw     1              ; LOW WORD OF TIMER COUNT
00018A9A <res 00000002>       <1> TIMER_HIGH:     resw     1              ; HIGH WORD OF TIMER COUNT
00018A9C <res 00000001>       <1> TIMER_OFL:      resb     1              ; TIMER HAS ROLLED OVER SINCE LAST READ
                              <1>
                              <1> ;----------------------------------------
                              <1> ;     DISKETTE DATA AREAS       :
                              <1> ;----------------------------------------
                              <1>
00018A9D <res 00000001>       <1> SEEK_STATUS: resb   1
00018A9E <res 00000001>       <1> MOTOR_STATUS:      resb   1
00018A9F <res 00000001>       <1> MOTOR_COUNT: resb   1
00018AA0 <res 00000001>       <1> DSKETTE_STATUS:    resb   1
00018AA1 <res 00000007>       <1> NEC_STATUS:  resb   7
                              <1>
                              <1> ;----------------------------------------
                              <1> ;     ADDITIONAL MEDIA DATA           :
                              <1> ;----------------------------------------
                              <1>
00018AA8 <res 00000001>       <1> LASTRATE:    resb   1
00018AA9 <res 00000001>       <1> HF_STATUS:   resb   1
00018AAA <res 00000001>       <1> HF_ERROR:    resb   1
00018AAB <res 00000001>       <1> HF_INT_FLAG: resb   1
00018AAC <res 00000001>       <1> HF_CNTRL:    resb   1
00018AAD <res 00000004>       <1> DSK_STATE:   resb   4
00018AB1 <res 00000002>       <1> DSK_TRK:     resb   2
                              <1>
                              <1> ;----------------------------------------
                              <1> ;     FIXED DISK DATA AREAS           :
                              <1> ;----------------------------------------
                              <1>
00018AB3 <res 00000001>       <1> DISK_STATUS1:      resb   1            ; FIXED DISK STATUS
00018AB4 <res 00000001>       <1> HF_NUM:            resb   1            ; COUNT OF FIXED DISK DRIVES
00018AB5 <res 00000001>       <1> CONTROL_BYTE:      resb   1            ; HEAD CONTROL BYTE
                              <1> ;@PORT_OFF   resb   1           ; RESERVED (PORT OFFSET)
                              <1> ;port1_off   resb   1           ; Hard disk controller 1 - port offset
```

```
                            <1> ;port2_off   resb   1            ; Hard idsk controller 2 - port offset
                            <1>
00018AB6 <res 00000002>     <1> alignb 4
                            <1>
                            <1> ;HF_TBL_VEC: resd   1            ; Primary master disk param. tbl. pointer
                            <1> ;HF1_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
                            <1> HF_TBL_VEC: ; 22/12/2014
00018AB8 <res 00000004>     <1> HDPM_TBL_VEC:      resd   1            ; Primary master disk param. tbl. pointer
00018ABC <res 00000004>     <1> HDPS_TBL_VEC:      resd   1            ; Primary slave disk param. tbl. pointer
00018AC0 <res 00000004>     <1> HDSM_TBL_VEC:      resd   1            ; Secondary master disk param. tbl. pointer
00018AC4 <res 00000004>     <1> HDSS_TBL_VEC:      resd   1            ; Secondary slave disk param. tbl. pointer
                            <1>
                            <1> ; 03/01/2015
00018AC8 <res 00000001>     <1> LBAMode:           resb   1
                            <1>
                            <1> ; ****************************************************************************

                                ;;; Real Mode Data (10/07/2015 - BSS)

                                ;alignb 2

                                ; 10/01/2016
                                %include 'trdoskx.s'      ; UNINITIALIZED KERNEL (Logical Drive & FS) DATA
                            <1> ; ****************************************************************************
                            <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.2) - UNINITIALIZED DATA : trdoskx.s
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Last Update: 30/08/2020
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Beginning: 04/01/2016
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Assembler: NASM version 2.11 (trdos386.s)
                            <1> ; ----------------------------------------------------------------------------
                            <1> ; Derived from TRDOS Operating System v1.0 (8086) source code by Erdogan Tan
                            <1> ; TRDOS2.ASM (09/11/2011)
                            <1> ; ****************************************************************************
                            <1> ; DRV_INIT.ASM [26/09/2009] Last Update: 07/08/2011
                            <1> ; MAINPROG.ASM [17/01/2004] Last Update: 09/11/2011
                            <1> ; DIR.ASM      [17/01/2004] Last Update: 09/10/2011
                            <1> ; CMD_INTR.ASM [29/01/2005] Last update: 09/11/2011
                            <1> ; DRV_FAT.ASM  [07/07/2009] Last update: 21/08/2011
                            <1>
00018AC9 <res 00000003>     <1> alignb 4
                            <1>
                            <1> ; MAINPROG.ASM
00018ACC <res 00000004>     <1> MainProgCfg_FileSize:   resd 1 ; 14/04/2016
00018AD0 <res 00000004>     <1> MainProgCfg_LineOffset: resd 1 ; 14/04/2016
                            <1>
00018AD4 <res 00000004>     <1> Current_VolSerial: resd 1
                            <1>
00018AD8 <res 00000004>     <1> Current_Dir_FCluster: resd 1
                            <1>
00018ADC <res 00000001>     <1> Current_Dir_Level: resb 1
00018ADD <res 00000001>     <1> Current_FATType: resb 1
00018ADE <res 00000001>     <1> Current_Drv: resb 1
00018ADF <res 00000001>     <1> Current_Dir_Drv:   resb 1 ; '?'
00018AE0 <res 00000001>     <1>                    resb 1 ; ':'
00018AE1 <res 00000001>     <1> Current_Dir_Root:  resb 1 ; '/'
00018AE2 <res 0000005A>     <1> Current_Directory: resb 90
00018B3C <res 00000001>     <1> End_Of_Current_Dir_Str: resb 1
00018B3D <res 00000001>     <1> Current_Dir_StrLen: resb 1
                            <1>
00018B3E <res 00000001>     <1> CursorColumn:      resb 1
00018B3F <res 00000001>     <1> CmdArgStart:    resb 1
                            <1>
                            <1> ; 03/02/2016
00018B40 <res 0000004E>     <1> Remark:            resb 78
                            <1>
00018B8E <res 00000050>     <1> CommandBuffer:     resb 80
                            <1>
00018BDE <res 00000100>     <1> TextBuffer:  resb 256
                            <1>
                            <1> MasterBootBuff:
00018CDE <res 000001BE>     <1> MasterBootCode: resb 1BEh
00018E9C <res 00000040>     <1> PartitionTable: resb 64
00018EDC <res 00000002>     <1> MBIDCode: resw 1
                            <1>
                            <1> PTable_Buffer:
00018EDE <res 00000040>     <1> PTable_hd0: resb 64
00018F1E <res 00000040>     <1> PTable_hd1: resb 64
00018F5E <res 00000040>     <1> PTable_hd2: resb 64
00018F9E <res 00000040>     <1> PTable_hd3: resb 64
                            <1> ; 15/07/2020
                            <1> ;PTable_ep0: resb 64
                            <1> ;PTable_ep1: resb 64
                            <1> ;PTable_ep2: resb 64
                            <1> ;PTable_ep3: resb 64
                            <1>
                            <1> ; 13/08/2020
00018FDE <res 00000001>     <1> scount: resb 1 ; 16/05/2016 (diskio.s, 'int33h:')
00018FDF <res 00000001>     <1>       resb 1
00018FE0 <res 00000001>     <1>       resb 1
00018FE1 <res 00000001>     <1>       resb 1
                            <1>
00018FE2 <res 00000001>     <1> HD_LBA_yes: resb 1
00018FE3 <res 00000001>     <1> PP_Counter: resb 1
00018FE4 <res 00000001>     <1> EP_Counter: resb 1
                            <1> ; 13/08/2020
00018FE5 <res 00000001>     <1> LD_Counter: resb 1
                            <1>
                            <1> ; 30/08/2020
00018FE6 <res 00000004>     <1> MBR_EP_StartSector: resd 1
                            <1>             ; Extd partition start sector as in MBR
00018FEA <res 00000004>     <1> EP_StartSector: resd 1 ; next extd partition start sector
                            <1>       ; 15/07/2020
                            <1>               ;resd 1
```

```
 83                                     <1>                 ;resd 1
 84                                     <1>
 85                                     <1> ; 20/07/2020
 86 00018FEE <res 00000200>             <1> DOSBootSectorBuff: resb 512
 87                                     <1> ; 15/07/2020
 88                                     <1> ;DOSBootSectorBuff: resb 446 ; 1BEh
 89                                     <1> ;MiniPartitionTable: resb 64 ;  40h
 90                                     <1> ;MiniPartitionMagic: resw  1 ;  02h
 91                                     <1>
 92                                     <1> FAT_BuffDescriptor:
 93 000191EE <res 00000004>             <1> FAT_CurrentCluster: resd 1
 94 000191F2 <res 00000001>             <1> FAT_BuffValidData: resb 1
 95 000191F3 <res 00000001>             <1> FAT_BuffDrvName: resb 1
 96 000191F4 <res 00000002>             <1> FAT_BuffOffset: resw 1
 97 000191F6 <res 00000004>             <1> FAT_BuffSector: resd 1
 98                                     <1>
 99 000191FA <res 00000004>             <1> FAT_ClusterCounter: resd 1
100 000191FE <res 00000004>             <1> LastCluster: resd 1
101                                     <1>
102                                     <1> ; 16/05/2016
103                                     <1> ;; 18/03/2016 (TRDOS v2.0)
104                                     <1> ;ClusterBuffer_Valid: resb 1
105                                     <1>
106                                     <1> Dir_BuffDescriptor:
107 00019202 <res 00000001>             <1> DirBuff_DRV: resb 1
108 00019203 <res 00000001>             <1> DirBuff_FATType: resb 1
109 00019204 <res 00000001>             <1> DirBuff_ValidData: resb 1
110 00019205 <res 00000002>             <1> DirBuff_CurrentEntry: resw 1
111 00019207 <res 00000002>             <1> DirBuff_LastEntry: resw 1
112 00019209 <res 00000004>             <1> DirBuff_Cluster: resd 1
113 0001920D <res 00000002>             <1> DirBuffer_Size: resw 1
114                                     <1> ;DirBuff_EntryCounter: resw 1
115                                     <1>
116                                     <1> ; 01/02/2016
117                                     <1> ; these are on (real mode) segment 8000h and later
118                                     <1> ; FAT_Buffer:      resb 1536 ; 3 sectors
119                                     <1> ; Dir_Buffer:      resb 512*32
120                                     <1> ; Logical_DOSDisks:  resb 6656 ; 26 * 256 bytes
121                                     <1>
122                                     <1> ; 18/01/2016
123                                     <1>
124 0001920F <res 00000004>             <1> FreeClusterCount: resd 1
125                                     <1>
126 00019213 <res 00000004>             <1> VolSize_Unit1:   resd 1
127 00019217 <res 00000004>             <1> VolSize_Unit2:   resd 1
128                                     <1>
129 0001921B <res 00000004>             <1> Vol_Tot_Sec_Str_Start:        resd 1
130 0001921F <res 0000000A>             <1> Vol_Tot_Sec_Str:       resb 10
131 00019229 <res 00000001>             <1> Vol_Tot_Sec_Str_End:          resb 1
132 0001922A <res 00000001>             <1> resb 1
133 0001922B <res 00000004>             <1> Vol_Free_Sectors_Str_Start: resd 1
134 0001922F <res 0000000A>             <1> Vol_Free_Sectors_Str:         resb 10
135 00019239 <res 00000001>             <1> Vol_Free_Sectors_Str_End:   resb 1
136                                     <1>
137                                     <1> ; 10/02/2016
138 0001923A <res 00000001>             <1> RUN_CDRV: resb 1 ; CMD_INTR.ASM  ; 09/11/2011
139                                     <1>
140                                     <1> ; 24/01/2016
141 0001923B <res 00000080>             <1> PATH_Array:     resb 128 ; DIR.ASM ; 09/10/2011
142                                     <1> ; 06/02/2016
143 000192BB <res 00000004>             <1> CCD_DriveDT: resd 1 ; DIR.ASM ; (word)
144 000192BF <res 00000001>             <1> CCD_Level:   resb 1 ; DIR.ASM
145 000192C0 <res 00000001>             <1> Last_Dir_Level:    resb 1 ; DIR.ASM
146                                     <1> ;
147 000192C1 <res 00000002>             <1> CDLF_AttributesMask: resw 1 ; DIR.ASM
148 000192C3 <res 00000004>             <1> CDLF_FNAddress:    resd 1 ; DIR.ASM (word)
149 000192C7 <res 00000002>             <1> CDLF_DEType: resw 1 ; DIR.ASM
150                                     <1> ;
151 000192C9 <res 00000001>             <1> CD_COMMAND:  resb 1 ; DIR.ASM
152                                     <1>
153 000192CA <res 00000002>             <1> alignb 4
154                                     <1>
155                                     <1> ; 29/01/2016
156 000192CC <res 00000001>             <1> Program_Exit:      resb 1 ; CMD_INTR.ASM  ; 09/11/2011
157                                     <1>
158                                     <1> ;alignb 4
159                                     <1> ; 23/02/2016
160 000192CD <res 00000001>             <1> disk_rw_op:  resb 1 ;  0 = disk read, 1 = disk write
161                                     <1> ;disk_rw_spt:      resb 1 ; sectors per track (<= 63) /// (<256)
162                                     <1> ; 31/01/2016
163 000192CE <res 00000001>             <1> retry_count:       resb 1 ; DISK_IO.ASM ; 20/07/2011 (CHS_RetryCount)
164 000192CF <res 00000001>             <1> disk_rw_err:       resb 1 ; DISK_IO.ASM ; (Disk_IO_err_code)
165 000192D0 <res 00000004>             <1> sector_count:      resd 1 ; DISK_IO.ASM ; (Disk_RW_SectorCount)
166                                     <1>
167                                     <1> ; 06/02/2016 (long name)
168 000192D4 <res 00000002>             <1> FDE_AttrMask:         resw 1 ; DIR.ASM
169 000192D6 <res 00000002>             <1> AmbiguousFileName: resw 1 ; DIR.ASM
170 000192D8 <res 00000001>             <1> PreviousAttr:         resb 1 ; DIR.ASM
171                                     <1> ;
172 000192D9 <res 00000001>             <1> LongNameFound:   resb 1     ; DIR.ASM
173 000192DA <res 00000001>             <1> LFN_EntryLength: resb 1   ; DIR.ASM
174 000192DB <res 00000001>             <1> LFN_CheckSum:    resb 1   ; DIR.ASM
175 000192DC <res 00000084>             <1> LongFileName:    resb 132 ; DIR.ASM
176                                     <1>
177                                     <1> ;PATH_Array_Ptr: resw 1 ; DIR.ASM
178 00019360 <res 00000001>             <1> PATH_CDLevel:       resb 1 ; DIR.ASM
179 00019361 <res 00000001>             <1> PATH_Level:   resb 1 ; DIR.ASM
180                                     <1>
181                                     <1> ; 07/02/2016
182 00019362 <res 0000000D>             <1> Dir_File_Name:     resb 13 ; DIR.ASM ; 09/10/2011
183                                     <1>
184                                     <1> ; 10/02/2016
185 0001936F <res 0000000D>             <1> Dir_Entry_Name:    resb 13 ; DIR.ASM
186                                     <1>
187                                     <1> alignb 2
```

```
                            <1>
0001937C <res 00000002>     <1> AttributesMask: resw 1 ; CMD_INTR.ASM ; 09/11/2011
                            <1>
                            <1> ; 10/02/2016 (128 bytes -> 126 bytes)
                            <1> ; 08/02/2016
                            <1> ;FFF Structure (128 bytes) ; DIR.ASM ; 09/10/2011
0001937E <res 00000001>     <1> FindFile_Drv:              resb 1
0001937F <res 00000041>     <1> FindFile_Directory:        resb 65
000193C0 <res 0000000D>     <1> FindFile_Name:             resb 13
                            <1> FindFile_LongNameEntryLength:
000193CD <res 00000001>     <1> FindFile_LongNameYes:      resb 1 ; Sign for longname procedures
                            <1> ;Above 80 bytes form
                            <1> ;TR-DOS Source/Destination File FullName Format/Structure
000193CE <res 00000002>     <1> FindFile_AttributesMask:  resw 1
000193D0 <res 00000020>     <1> FindFile_DirEntry:   resb 32
000193F0 <res 00000004>     <1> FindFile_DirFirstCluster: resd 1
000193F4 <res 00000004>     <1> FindFile_DirCluster:       resd 1
000193F8 <res 00000002>     <1> FindFile_DirEntryNumber:  resw 1
000193FA <res 00000002>     <1> FindFile_MatchCounter:     resw 1
000193FC <res 00000002>     <1> FindFile_Reserved:   resw 1 ; 06/03/2016
                            <1>
000193FE <res 00000004>     <1> First_Path_Pos: resd 1    ; DIR.ASM ; 09/10/2011
00019402 <res 00000004>     <1> Last_Slash_Pos: resd 1    ; DIR.ASM
                            <1>
                            <1> ; 10/02/2016
00019406 <res 00000002>     <1> File_Count:     resw 1    ; DIR.ASM ; 09/10/2011
00019408 <res 00000002>     <1> Dir_Count:      resw 1
0001940A <res 00000004>     <1> Total_FSize:    resd 1
0001940E <res 00000004>     <1> TFS_Dec_Begin:  resd 1
00019412 <res 0000000A>     <1>                 resb 10
0001941C <res 00000001>     <1> TFS_Dec_End:    resb 1
                            <1>
0001941D <res 00000001>     <1> PrintDir_RowCounter: resb 1
                            <1>
0001941E <res 00000002>     <1> alignb 4
                            <1> ; 15/02/2015 ('show' command variables)
00019420 <res 00000004>     <1> Show_FDT:    resd 1
00019424 <res 00000004>     <1> Show_LDDDT:  resd 1
00019428 <res 00000004>     <1> Show_Cluster:      resd 1
0001942C <res 00000004>     <1> Show_FileSize:     resd 1
00019430 <res 00000004>     <1> Show_FilePointer: resd 1
00019434 <res 00000002>     <1> Show_ClusterPointer: resw 1
00019436 <res 00000002>     <1> Show_ClusterSize: resw 1
00019438 <res 00000001>     <1> Show_RowCount:     resb 1
                            <1>
00019439 <res 00000003>     <1> alignb 4
                            <1> ; 21/02/2016
0001943C <res 00000004>     <1> DelFile_FNPointer: resd 1 ; ; CMD_INTR.ASM (word) ; 09/11/2011
                            <1> ; 27/02/2016
                            <1> ; DIR.ASM (09/10/2011)
00019440 <res 00000004>     <1> DelFile_FCluster:  resd 1
00019444 <res 00000002>     <1> DelFile_EntryCounter:     resw 1
00019446 <res 00000001>     <1> DelFile_LNEL:             resb 1
00019447 <res 00000001>     <1> resb 1
                            <1>
                            <1> ; DIR.ASM
00019448 <res 00000004>     <1> mkdir_DirName_Offset:     resd 1
0001944C <res 00000004>     <1> mkdir_FFCluster:   resd 1
00019450 <res 00000004>     <1> mkdir_LastDirCluster:     resd 1
00019454 <res 00000004>     <1> mkdir_FreeSectors: resd 1
00019458 <res 00000002>     <1> mkdir_attrib:             resw 1
0001945A <res 00000001>     <1> mkdir_SecPerClust: resb 1
0001945B <res 00000001>     <1> mkdir_add_new_cluster:    resb 1
0001945C <res 0000000D>     <1> mkdir_Name:        resb 13
00019469 <res 00000002>     <1> resw 1 ; 01/03/2016
                            <1> ; 27/02/2016
0001946B <res 00000001>     <1> RmDir_MultiClusters:      resb 1
0001946C <res 00000004>     <1> RmDir_DirEntryOffset:     resd 1 ; 01/03/2016 (word -> dword)
00019470 <res 00000004>     <1> RmDir_ParentDirCluster: resd 1
00019474 <res 00000004>     <1> RmDir_DirLastCluster:   resd 1
00019478 <res 00000004>     <1> RmDir_PreviousCluster:  resd 1
                            <1> ; 22/02/2016
0001947C <res 00000001>     <1> UPDLMDT_CDirLevel: resb 1
0001947D <res 00000004>     <1> UPDLMDT_CDirFCluster:     resd 1
                            <1>
00019481 <res 00000003>     <1> alignb 4
                            <1> ; DRV_FAT.ASM ; 21/08/2011
00019484 <res 00000004>     <1> gffc_next_free_cluster:  resd 1
00019488 <res 00000004>     <1> gffc_first_free_cluster: resd 1
0001948C <res 00000004>     <1> gffc_last_free_cluster:  resd 1
                            <1>
                            <1> ;29/04/2016
                            <1> Cluster_Index: ; resd 1
                            <1> ; 22/02/2016
00019490 <res 00000004>     <1> ClusterValue:      resd 1
                            <1> ; 04/03/2016
00019494 <res 00000001>     <1> Attributes: resb 1
                            <1> ;;CFS_error:  resb 1 ;; 01/03/2016
00019495 <res 00000001>     <1> resb 1
00019496 <res 00000001>     <1> CFS_OPType: resb 1
00019497 <res 00000001>     <1> CFS_Drv:    resb 1
00019498 <res 00000004>     <1> CFS_CC:         resd 1
0001949C <res 00000004>     <1> CFS_FAT32FSINFOSEC: resd 1
000194A0 <res 00000004>     <1> CFS_FAT32FC: resd 1
                            <1>
                            <1> ; 27/02/2016
                            <1> ;alignb 4
000194A4 <res 00000004>     <1> glc_prevcluster: resd 1 ; DRV_FAT.ASM (21/08/2011)
                            <1> ; 22/10/2016
000194A8 <res 00000004>     <1> glc_index:   resd 1 ;  Last Cluster Index (22/10/2016)
                            <1>
                            <1> ; DIR.ASM
000194AC <res 00000002>     <1> DLN_EntryNumber: resw 1
000194AE <res 00000001>     <1> DLN_40h:     resb 1
                            <1> ; 28/02/2016
```

```
000194AF <res 00000001>      <1> TCC_FATErr:   resb 1 ; DRV_FAT.ASM
                             <1>
                             <1> alignb 4
                             <1> ; DIR.ASM (09/10/2011)
000194B0 <res 00000002>      <1> LCDE_EntryIndex: resw 1 ; LCDE_EntryOffset
000194B2 <res 00000002>      <1> LCDE_ClusterSN:  resw 1
000194B4 <res 00000004>      <1> LCDE_Cluster:       resd 1
000194B8 <res 00000004>      <1> LCDE_ByteOffset: resd 1
                             <1>
                             <1> ;alignb4
                             <1> ; 06/03/2016 (word -> dword)
                             <1> ; CMD_INTR.ASM (01/08/2010)
000194BC <res 00000004>      <1> SourceFilePath:         resd 1
000194C0 <res 00000004>      <1> DestinationFilePath: resd 1
                             <1>
                             <1> ;alignb 4
                             <1> ; 06/03/2016
                             <1> ; FILE.ASM (09/10/2011)
                             <1> ;Source File Structure (same with 'Find File' Structure)
000194C4 <res 00000001>      <1> SourceFile_Drv:                 resb 1
000194C5 <res 00000041>      <1> SourceFile_Directory:           resb 65
00019506 <res 0000000D>      <1> SourceFile_Name:          resb 13
                             <1> SourceFile_LongNameEntryLength:
00019513 <res 00000001>      <1> SourceFile_LongNameYes:         resb 1 ; Sign for longname procedures
                             <1> ;Above 80 bytes
                             <1> ;is TR-DOS Source File FullName Format/Structure
00019514 <res 00000002>      <1> SourceFile_AttributesMask:      resw 1
00019516 <res 00000020>      <1> SourceFile_DirEntry:            resb 32
00019536 <res 00000004>      <1> SourceFile_DirFirstCluster:     resd 1
0001953A <res 00000004>      <1> SourceFile_DirCluster:          resd 1
0001953E <res 00000002>      <1> SourceFile_DirEntryNumber:      resw 1
00019540 <res 00000002>      <1> SourceFile_MatchCounter:  resw 1
                             <1> ; 16/03/2016
00019542 <res 00000001>      <1> SourceFile_SecPerClust:         resb 1
00019543 <res 00000001>      <1> SourceFile_Reserved:            resb 1
                             <1> ; Above is 128 bytes
                             <1>
                             <1> ;Destination File Structure (same with 'Find File' Structure)
00019544 <res 00000001>      <1> DestinationFile_Drv:            resb 1
00019545 <res 00000041>      <1> DestinationFile_Directory:      resb 65
00019586 <res 0000000D>      <1> DestinationFile_Name:           resb 13
                             <1> DestinationFile_LongNameEntryLength:
00019593 <res 00000001>      <1> DestinationFile_LongNameYes:    resb 1 ; Sign for longname procedures
                             <1> ;Above 80 bytes
                             <1> ;is TR-DOS Destination File FullName Format/Structure
00019594 <res 00000002>      <1> DestinationFile_AttributesMask: resw 1
00019596 <res 00000020>      <1> DestinationFile_DirEntry: resb 32
000195B6 <res 00000004>      <1> DestinationFile_DirFirstCluster: resd 1
000195BA <res 00000004>      <1> DestinationFile_DirCluster:     resd 1
000195BE <res 00000002>      <1> DestinationFile_DirEntryNumber: resw 1
000195C0 <res 00000002>      <1> DestinationFile_MatchCounter:   resw 1
                             <1> ; 16/03/2016
000195C2 <res 00000001>      <1> DestinationFile_SecPerClust:    resb 1
000195C3 <res 00000001>      <1> DestinationFile_Reserved: resb 1
                             <1> ; Above is 128 bytes
                             <1>
                             <1> ; 24/04/2016
000195C4 <res 00000002>      <1> resw 1
                             <1>
                             <1> ; 10/03/2016
                             <1> ; FILE.ASM
000195C6 <res 00000001>      <1> move_cmd_phase:       resb 1
000195C7 <res 00000001>      <1> msftdf_sf_df_drv:  resb 1
000195C8 <res 00000004>      <1> msftdf_drv_offset: resd 1
                             <1>
                             <1> ; 11/03/2016
                             <1> ; DRV_FAT.ASM (21/08/2011)
000195CC <res 00000004>      <1> FAT_anc_LCluster:  resd 1
000195D0 <res 00000004>      <1> FAT_anc_FFCluster: resd 1
                             <1>
                             <1> ;alignb 4
                             <1>
                             <1> ; 14/03/2016
                             <1> ; TRDOS 386 = TRDOS v2.0 feature only !
                             <1> ; 'allocate_memory_block' in 'memory.s'
000195D4 <res 00000004>      <1> mem_ipg_count:     resd 1 ; page count (for contiguous allocation)
000195D8 <res 00000004>      <1> mem_pg_count:      resd 1 ; page count (for count down)
000195DC <res 00000004>      <1> mem_aperture:      resd 1 ; contiguous free pages (current)
000195E0 <res 00000004>      <1> mem_max_aperture: resd 1 ; maximum value of contiguous free pages
000195E4 <res 00000004>      <1> mem_pg_pos:  resd 1 ; mem. position (page #) of current aperture
000195E8 <res 00000004>      <1> mem_max_pg_pos: resd 1 ; mem. position (page #) of max. aperture
                             <1>
                             <1> ; 15/03/2016
                             <1> ; FILE.ASM ('copy_source_file_to_destination_file')
000195EC <res 00000001>      <1> copy_cmd_phase:       resb 1
000195ED <res 00000001>      <1> csftdf_rw_err:           resb 1
000195EE <res 00000001>      <1> DestinationFileFound: resb 1
000195EF <res 00000001>      <1> csftdf_cdrv:             resb 1
000195F0 <res 00000004>      <1> csftdf_filesize:      resd 1
                             <1> ; TRDOS386 (TRDOS v2.0)
000195F4 <res 00000004>      <1> csftdf_sf_mem_addr:   resd 1
000195F8 <res 00000004>      <1> csftdf_sf_mem_bsize:  resd 1
                             <1> ;
                             <1>
000195FC <res 00000004>      <1> csftdf_sf_cluster:    resd 1 ; 16/03/2016
00019600 <res 00000004>      <1> csftdf_df_cluster:    resd 1
                             <1> ; 16/03/2016
00019604 <res 00000004>      <1> csftdf_r_size:        resd 1
00019608 <res 00000004>      <1> csftdf_w_size:        resd 1
0001960C <res 00000004>      <1> csftdf_sf_rbytes:     resd 1
00019610 <res 00000004>      <1> csftdf_df_wbytes:     resd 1
00019614 <res 00000001>      <1> csftdf_percentage:    resb 1
                             <1> ; 17/03/2016
00019615 <res 00000001>      <1> csftdf_videopage:     resb 1
00019616 <res 00000002>      <1> csftdf_cursorpos:     resw 1
```

```
00019618 <res 00000004>      <1> csftdf_sf_drv_dt:     resd 1
0001961C <res 00000004>      <1> csftdf_df_drv_dt:     resd 1
                             <1>
                             <1> ; 21/03/2016
                             <1> ; 20/03/2016
                             <1> ; FILE.ASM
00019620 <res 00000004>      <1> createfile_Name_Offset:  resd 1
00019624 <res 00000004>      <1> createfile_FreeSectors:  resd 1
00019628 <res 00000004>      <1> createfile_size:         resd 1
0001962C <res 00000004>      <1> createfile_FFCluster:    resd 1 ; 11/03/2016
00019630 <res 00000004>      <1> createfile_LastDirCluster: resd 1
00019634 <res 00000004>      <1> createfile_Cluster:      resd 1
00019638 <res 00000004>      <1> createfile_PCluster:     resd 1
0001963C <res 00000001>      <1> createfile_attrib:  resb 1
0001963D <res 00000001>      <1> createfile_SecPerClust:  resb 1
0001963E <res 00000002>      <1> createfile_DirIndex:     resw 1
00019640 <res 00000004>      <1> createfile_CCount:  resd 1
00019644 <res 00000002>      <1> createfile_BytesPerSec:    resw 1 ; 23/03/2016
00019646 <res 00000001>      <1> createfile_wfc:              resb 1
00019647 <res 00000001>      <1> createfile_UpdatePDir:     resb 1 ; 31/03/2016
                             <1>
                             <1> ;alignb 4
                             <1>
                             <1> ; 11/04/2016
00019648 <res 00000002>      <1> env_var_length:       resw 1
                             <1>
0001964A <res 00000002>      <1> alignb 4
                             <1>
                             <1> ; 25/04/2016
0001964C <res 00000001>      <1> readi.valid: resb 1 ; valid data (>0 = valid for readi)
0001964D <res 00000001>      <1> readi.drv:   resb 1 ; drive number (0, 1,2,3,4..)
0001964E <res 00000001>      <1> readi.spc:   resb 1 ; sectors per cluster for 'readi' drive
0001964F <res 00000001>      <1> readi.s_index:  resb 1 ; sector index in current cluster (buffer)
00019650 <res 00000004>      <1> readi.sector:      resd 1 ; current disk sector
00019654 <res 00000002>      <1> readi.bpc:  resw 1 ; bytes per cluster - 1
00019656 <res 00000002>      <1> readi.offset:      resw 1 ; byte offset in cluster buffer
00019658 <res 00000004>      <1> readi.cluster:  resd 1 ; current cluster number
0001965C <res 00000004>      <1> readi.c_index:     resd 1 ; cluster index of the current cluster (0,1,2,3..)
00019660 <res 00000004>      <1> readi.fclust:      resd 1 ; first cluster of the current cluster
00019664 <res 00000004>      <1> readi.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
                             <1> ;readi.buffer:     resd 1 ; readi sector buffer address
                             <1>
                             <1> ;alignb 4
                             <1>
00019668 <res 00000001>      <1> writei.valid:      resb 1 ; valid data (>0 = valid for writei)
00019669 <res 00000001>      <1> writei.drv: resb 1 ; drive number (0, 1,2,3,4..)
0001966A <res 00000001>      <1> writei.spc: resb 1 ; sectors per cluster for 'writei' drive
0001966B <res 00000001>      <1> writei.s_index: resb 1 ; sector index in current cluster (buffer)
0001966C <res 00000004>      <1> writei.sector:     resd 1 ; current disk sector
00019670 <res 00000002>      <1> writei.bpc: resw 1 ; bytes per cluster - 1
00019672 <res 00000002>      <1> writei.offset:     resw 1 ; byte offset in cluster buffer
00019674 <res 00000004>      <1> writei.cluster: resd 1 ; current cluster number
00019678 <res 00000004>      <1> writei.c_index:    resd 1 ; cluster index of the current cluster (0,1,2,3..)
0001967C <res 00000004>      <1> writei.fclust:  resd 1 ; first cluster of the current cluster
00019680 <res 00000004>      <1> writei.fs_index: resd 1 ; sector index in disk/file section (for Singlix FS)
                             <1> ;writei.buffer:    resd 1 ; writei sector buffer address
00019684 <res 00000004>      <1> writei.lclust:     resd 1 ; writei last cluster (mget_w) ; 23/10/2016
00019688 <res 00000004>      <1> writei.l_index:    resd 1 ; writei last cluster index (mget_w) ; 23/10/2016
0001968C <res 00000001>      <1> writei.ofn: resb 1 ; open file number (to be written) ; 23/10/2016
                             <1>
0001968D <res 00000003>      <1> alignb 4
                             <1>
                             <1> ; 29/04/2016
00019690 <res 00000004>      <1> Run_CDirFC: resd 1
00019694 <res 00000001>      <1> Run_Auto_Path:     resb 1
00019695 <res 00000001>      <1> Run_Manual_Path: resb 1 ; 0 -> auto path sequence needed
00019696 <res 00000001>      <1> EXE_ID:            resb 1
00019697 <res 00000001>      <1> EXE_dot:    resb 1
                             <1>
                             <1> ; 06/05/2016
00019698 <res 00000004>      <1> mainprog_return_addr: resd 1
0001969C <res 00000004>      <1> last_error:  resd 1  ; this will be used to return error code to MainProg
                             <1>                      ; 'lasterror' keyword will be used later to get the
                             <1>                      ; last error code/number/status.
                             <1> ; 12/05/2016
000196A0 <res 00000004>      <1> video_eax:   resd 1  ; eax return value of video function
                             <1>
                             <1> ; 01/06/2016
000196A4 <res 00000004>      <1> user_buffer: resd 1  ; 'diskio.s' (INT 33h, Function 08h, floppy disk type)
                             <1>
                             <1> ; 21/05/2016 - TRDOS 386 ('swap/switch', 'rswap', [u.pri])
000196A8 <res 00000001>      <1> priority:    resb 1  ; running priority level of process (0,1,2)
                             <1>                      ; (run queue which is process comes from)
                             <1> ; 22/05/2016 - TRDOS 386 ('set_run_sequence', 'rtc_int', 'u_timer')
000196A9 <res 00000001>      <1> p_change:    resb 1  ; process change status (for timer events)
                             <1> ; 23/05/2016 - TRDOS 386 ('clock')
000196AA <res 00000001>      <1> multi_tasking:     resb 1   ; Multi Tasking status (0 = disabled, >0 = enabled)
                             <1>                      ; (EBX will return with user buffer addr or disk type)
                             <1> ; 07/06/2016
000196AB <res 00000001>      <1> timer_events:      resb 1  ; number of (active) timer events, <= 16
                             <1>
                             <1> ; 24/06/2016
000196AC <res 00000001>      <1> w_str_cmd:   resb 1 ; WRITE_STRING command (0,1,2,3) ; video.s
000196AD <res 00000001>      <1> p_crt_mode:  resb 1  ; previous video mode (=3 or 0), backup mark/sign
                             <1> ; 26/06/2016
000196AE <res 00000001>      <1> p_crt_page:  resb 1  ; previous active page (for 'set_mode')
                             <1> ; 04/07/2016
000196AF <res 00000001>      <1> noclearmem:  resb 1  ; if set, 'SET MODE' (INT 31h) function (AH = 4)
                             <1>                      ; will not clear the video memory
                             <1>                      ; (usable for graphics modes only)
                             <1> alignb 2
000196B0 <res 00000002>      <1> CRT_LEN:     resw 1  ; length of regen buffer in bytes
000196B2 <res 00000010>      <1> cursor_pposn:      resw 8  ; cursor positions backup
                             <1>
                             <1> ; 10/07/2016 ('VGA_FONT_SETUP', INT 43H address for x86 real mode bios)
```

```
000196C2 <res 00000004>      <1> VGA_INT43H:  resd 1 ; 0 = default (not configured by user)
                             <1>                     ; 0FFFFFFFFh = user defined fonts
                             <1>                     ; address:
                             <1>                     ;      vgafont8
                             <1>                     ;      vgafont16
                             <1>                     ;      vgafont14
                             <1>
                             <1> ; 25/07/2016
000196C6 <res 00000001>      <1> VGA_MTYPE:   resb 1  ; 0=CTEXT,1=MTEXT,2=CGA,3=PLANAR1,4=PLANAR4,5=LINEAR
                             <1>
                             <1> ; 23/10/2016
000196C7 <res 00000001>      <1> setfmod             resb 1 ; update last modification date&time sign (if >0)
                             <1>                     ; (it is Open File Number + 1, if > 0)
                             <1> alignb 4
                             <1>
                             <1> ; 16/10/2016
000196C8 <res 00000004>      <1> FFF_UBuffer: resd 1  ; User's buffer address for FFF & FNF system calls
                             <1> ; 15/10/2016
000196CC <res 00000001>      <1> FFF_Valid:   resb 1  ; Find First File Structure validation byte
                             <1>                     ; 0  = invalid (Find Next File can't use FFF struct)
                             <1>                     ; >0 = valid, return type for FFF and Find Next File
                             <1>                     ; 24 = basic parameters, 24 bytes
                             <1>                     ; 128 = entire FFF structure/table, 128 bytes
                             <1> ; 16/10/2016 (FFF_Attrib: resw 1)
000196CD <res 00000001>      <1> FFF_Attrib:  resb 1 ; Find First File attributes for Find Next File (LB)
000196CE <res 00000001>      <1> FFF_RType:   resb 1  ; FFF return type (0 = Basic, >0 = complete) (HB)
                             <1> ; 16/10/2016 - 05/10/2016 (Set Working Path)
000196CF <res 00000001>      <1> SWP_inv_fname:     resb 1 ; Set Working Path - Invalid File Name
000196D0 <res 00000002>      <1> SWP_Mode:    resw 1 ; Set Working Path - Mode
000196D2 <res 00000001>      <1> SWP_DRV:     resb 1 ; Set Working Path - Drive
000196D3 <res 00000001>      <1> SWP_DRV_chg: resb 1 ; Set Working Path - Drive Change
                             <1>
                             <1> ; 27/02/2017
000196D4 <res 00000001>      <1> fpready:     resb 1 ; '80387 fpu is ready' flag
                             <1>
                             <1> ; 08/10/2016
000196D5 <res 00000009>      <1> device_name:    resb 9  ; capitalized (and zero padded) device canem
                             <1>                    ; (example: "TTY0",0,0,0,0,0")
                             <1>
000196DE <res 00000002>      <1> alignb 4
                             <1>
                             <1> ; 08/10/2016
                             <1> ; 07/10/2016
                             <1> ; Table of kernel devices (which do not use installable device drivers)
                             <1> ; has been coded into KERNEL (trdosk9.s)
                             <1> ; 07/10/2016
                             <1> ; 8 installable device drivers available to install (NUMIDEV)
000196E0 <res 00000020>      <1> IDEV_PGDIR: resd NUMIDEV
                             <1>                     ; Page directories of installable device drivers
                             <1>                     ;
                             <1>                     ; Note: Virtual start address is always 400000h
                             <1>                     ; (end of the 1st 4MB). [org 400000h]
                             <1>                     ; Segments: KCODE, KDATA
                             <1>                     ; Method: call 400000h (after changing page dir)
                             <1>                     ; Query code located at the start (400000h).
                             <1>                     ; Query code returns with
                             <1>                     ;   eax = device type and driver version
                             <1>                     ;         AL = Device Type minor
                             <1>                     ;         AH = Device Type major
                             <1>                     ;         Byte 16-23 : Version minor
                             <1>                     ;       Byte 24-31 : Version major - 1
                             <1>                     ;                    (0:0 -> 1.0)
                             <1>                     ;   ebx = initialization code address
                             <1>                     ;   ecx = configuration table address
                             <1>                     ;   edx = description table address
                             <1>                     ;   esi = device (default) name address (ASCIIZ)
                             <1>                     ;       (name has "/DEV/" prefix)
                             <1>                     ;   edi = dispatch table address
                             <1>                     ;        (for calling kernel-device functions)
                             <1>                     ;   ebp = address table address
                             <1>                     ; Initialization code returns with
                             <1>                     ;   eax = open code address
                             <1>                     ;   ecx = close code address
                             <1>                     ;   ebx = read code address
                             <1>                     ;   edx = write code address
                             <1>                     ;   esi = IOCTL code address
                             <1>                     ;   edi = dispatch table address
                             <1>                     ;   ebp = address table address
                             <1>                     ; Address Table:
                             <1>                     ;    Offset 0  : open code address
                             <1>                     ;    Offset 4  : read code address
                             <1>                     ;    Offset 8  : write code address
                             <1>                     ;    Offset 12 : close code address
                             <1>                     ;    Offset 16 : IOCTL code address
                             <1>                     ;    Offset 20 : initialization code address
                             <1>                     ;    Offset 24 : description table address
                             <1>                     ;    Offset 28 : configuration table address
                             <1>                     ;    Offset 32 : device name address
                             <1>                     ;    Offset 36 : dispatch table address
                             <1>                     ;          (for calling kernel-device functions)
                             <1>
00019700 <res 00000040>      <1> IDEV_NAME:  resb 8*NUMIDEV
                             <1>                     ; 8 byte names of installable device drivers
                             <1>
00019740 <res 00000008>      <1> IDEV_TYPE:  resb NUMIDEV ; Driver type of installable device drivers
00019748 <res 00000008>      <1> IDEV_FLAGS: resb NUMIDEV ; Device access parameters for installable
                             <1>                          ; device drivers (These values are set while
                             <1>                       ; the device driver is being loaded.)
00019750 <res 00000020>      <1> IDEV_OADDR: resd NUMIDEV ; open function addr for installable dev driver
00019770 <res 00000020>      <1> IDEV_CADDR: resd NUMIDEV ; close function addr for installable dev driver
00019790 <res 00000020>      <1> IDEV_RADDR: resd NUMIDEV ; read function addr for installable dev driver
000197B0 <res 00000020>      <1> IDEV_WADDR: resd NUMIDEV ; write function addr for installable dev driver
                             <1>
                             <1> ; 08/10/2016
                             <1> ; 07/10/2016
```

```
                            <1> ; Device Open and Access parameters
000197D0 <res 0000001E>     <1> DEV_ACCESS:   resb NUMOFDEVICES   ; bit 0 = accessable by normal users
                            <1>                                   ; bit 1 = read access permission
                            <1>                                   ; bit 2 = write access permission
                            <1>                                   ; bit 3 = IOCTL permission to users
                            <1>                                   ; bit 4 = block device if it is set
                            <1>                                   ; bit 5 = 16 bit or 1024 byte data
                            <1>                                   ; bit 6 = 32 bit or 2048 byte data
                            <1>                                   ; bit 7 = installable device driver
000197EE <res 0000001E>     <1> DEV_R_OWNER:  resb NUMOFDEVICES   ; Reading owner no (u.uid) of devices
0001980C <res 0000001E>     <1> DEV_R_OPENCOUNT: resb NUMOFDEVICES   ; Reading open count
0001982A <res 0000001E>     <1> DEV_W_OWNER:  resb NUMOFDEVICES   ; Writing owner no (u.uid) of devices
00019848 <res 0000001E>     <1> DEV_W_OPENCOUNT: resb NUMOFDEVICES   ; Writing open count
00019866 <res 0000001E>     <1> DEV_DRIVER:   resb NUMOFDEVICES   ; device driver number (1 to 7Fh)
                            <1>                                   ; *if bit 7 is set (80 to FFh)
                            <1>                                   ; *if it is installable device driver
                            <1>                                   ; *index (0 to 7Fh)
                            <1>                                   ; otherwise it is kernel device index
00019884 <res 0000001E>     <1> DEV_OPENMODE:       resb NUMOFDEVICES   ; 1 = read mode
                            <1>                                   ; 2 = write mode
                            <1>                                   ; 3 = read & write
                            <1>                                   ; 0 = not open (free)
000198A2 <res 00000078>     <1> DEV_NAME_PTR:    resd NUMOFDEVICES   ; pointers to name addresses of drivers
                            <1>                                   ; Address base: KDEV_NAME+
                            <1>                                   ; or IDEV_NAME+
0001991A <res 00000078>     <1> DEV_R_POINTER:      resd NUMOFDEVICES   ; reading pointer, writing pointer
00019992 <res 00000078>     <1> DEV_W_POINTER:      resd NUMOFDEVICES   ; sector number if block device
                            <1>                                   ; character offset if char device
00019A0A <res 00000002>     <1> alignb 4
                            <1>
                            <1> ; 06/10/2016
                            <1> ; Open File Parameters
00019A0C <res 00000028>     <1> OF_FCLUSTER: resd OPENFILES  ; First clusters of open files
00019A34 <res 0000000A>     <1> OF_DRIVE:    resb OPENFILES  ; Logical DOS drive numbers of open files
00019A3E <res 0000000A>     <1> OF_MODE:     resb OPENFILES  ; Open mode (1 = read, 2 = write, 3 = r&w)
00019A48 <res 0000000A>     <1> OF_STATUS:   resb OPENFILES  ; (bit 0 = read, bit 1 = write)
00019A52 <res 0000000A>     <1> OF_OPENCOUNT:      resb OPENFILES  ; Open counts of open files
00019A5C <res 00000028>     <1> OF_POINTER:  resd OPENFILES      ; File seek/read/write pointer
00019A84 <res 00000028>     <1> OF_SIZE:     resd OPENFILES      ; File sizes of open files (in bytes)
00019AAC <res 00000028>     <1> OF_DIRFCLUSTER:    resd OPENFILES  ; Directory First Clusters of open files
00019AD4 <res 00000028>     <1> OF_DIRCLUSTER:     resd OPENFILES  ; Directory (Entry) Clusters of open files
00019AFC <res 00000028>     <1> OF_VOLUMEID: resd OPENFILES  ; Vol ID for removable drives of open files
00019B24 <res 00000028>     <1> OF_CCLUSTER: resd OPENFILES  ; Current clusters of open files
00019B4C <res 00000028>     <1> OF_CCINDEX:  resd OPENFILES  ; Cluster index numbers of current clusters
                            <1> ; 24/10/2016
00019B74 <res 00000014>     <1> OF_DIRENTRY: resw OPENFILES  ; Directory entry index no. in dir cluster
                            <1>                              ; Sector index = entry index / 16
                            <1> ;alignb 2
                            <1>
00019B88 <res 00000060>     <1> DTA:         resd 24             ; Find First File data transfer area
                            <1>
                            <1> ; 19/12/2016
00019BE8 <res 00000001>     <1> tcallback:   resb 1       ; Timer callback method flag for 'systimer'
00019BE9 <res 00000001>     <1> trtc:        resb 1       ; Timer interrupt type flag for 'systimer'
                            <1> ; 20/02/2017
00019BEA <res 00000001>     <1> no_page_swap:      resb 1       ; Swap lock for Signal Response Byte pages
                            <1> ;;15/01/2017
                            <1> ; 02/01/2017
                            <1> ;;intflg:    resb 1       ; software interrupt in progress signal
                            <1>                           ; (for timer interrupt)
                            <1>
00019BEB <res 00000001>     <1> alignb 4
                            <1> ; 13/04/2017
00019BEC <res 0000001E>     <1> DEV_INTR:    resb NUMOFDEVICES ; Device Interrupt (IRQ) number + 1
                            <1>                           ; (0= not available, 1= IRQ 0, 16= IRQ 15)
00019C0A <res 00000040>     <1> DEV_INT_HNDLR:     resd 16            ; Device Interrupt Handler addr, if > 0
                            <1>
                            <1>
                            <1> ;alignb 4
                            <1>
                            <1> ; 26/02/2017 ; IRQ Callback parameters ('syscalbac')
                            <1> ;Index: ; 0 to 8
                            <1> ;     0 = IRQ3, 1 = IRQ4, 2 = IRQ5, 3 = IRQ7
                            <1> ;     4 = IRQ9, 5 = IRQ10, 6 = IRQ11, 7 = IRQ12, 8 = IRQ13
00019C4A <res 00000009>     <1> IRQ.owner:   resb 9       ; owner, 0 = free, >0 = [u.uno]
00019C53 <res 00000009>     <1> IRQ.dev:     resb 9       ; 0 = default/kernel, >0 = device number
00019C5C <res 00000009>     <1> IRQ.method:  resb 9            ; 0 = Signal Response Byte, 1 = Callback
00019C65 <res 00000009>     <1> IRQ.srb:     resb 9            ; Signal Response/Return Byte value
00019C6E <res 00000024>     <1> IRQ.addr:    resd 9       ; Rignal Response Byte address (physical)
                            <1>                           ; or Callback service address (virtual)
                            <1> ; 28/02/2017
00019C92 <res 00000004>     <1> IRQ_cr3:     resd 1       ; for saving cr3 register in IRQ handler
00019C96 <res 00000001>     <1> IRQnum:            resb 1       ; IRQ number for IRQ handler (trdosk8.s)
                            <1>
                            <1> ; 10/04/2017
                            <1> ; 03/04/2017
                            <1> ; UNINITIALIZED AUDIO DATA
00019C97 <res 00000001>     <1> alignb 4
00019C98 <res 00000001>     <1> audio_pci:   resb 1
00019C99 <res 00000001>     <1> audio_device:      resb 1
00019C9A <res 00000001>     <1> audio_mode:  resb 1
00019C9B <res 00000001>     <1> audio_intr:  resb 1
00019C9C <res 00000001>     <1> audio_busy:  resb 1  ; Busy flag for audio irq ; 21/04/2017
00019C9D <res 00000001>     <1> audio_reserved: resb 1
00019C9E <res 00000002>     <1> audio_io_base:     resw 1       ; Base I/O address of audio device
00019CA0 <res 00000004>     <1> audio_dev_id:      resd 1 ; BUS/DEV/FN ; 00000000BBBBBBBBDDDDDFFF00000000
00019CA4 <res 00000004>     <1> audio_vendor:      resd 1
00019CA8 <res 00000004>     <1> audio_stats_cmd: resd 1
                            <1> ;
00019CAC <res 00000004>     <1> audio_buffer:      resd 1 ; virtual address of user's audio buffer
00019CB0 <res 00000004>     <1> audio_p_buffer:    resd 1 ; Physical address of user's audio buffer
00019CB4 <res 00000004>     <1> audio_buff_size: resd 1 ; user's audio buffer size (half buffer size)
00019CB8 <res 00000004>     <1> audio_dma_buff: resd 1  ; dma buffer address
00019CBC <res 00000004>     <1> audio_dmabuff_size: resd 1 ; dma buffer size (2 * half buffer size)
00019CC0 <res 00000001>     <1> audio_flag:  resb 1  ; dma buffer flag (1st half = 0, 2nd half = 1)
```

```
00019CC1 <res 00000001>      <1> audio_user:  resb 1 ; user number of the owner
00019CC2 <res 00000001>      <1> audio_cb_mode:     resb 1 ; 0 = signal response byte method
                             <1>                    ; 1 = callback method
                             <1>                    ; 2 = s.r.b. method with auto increment
00019CC3 <res 00000001>      <1> audio_srb:   resb 1 ; signal response byte value
00019CC4 <res 00000004>      <1> audio_cb_addr:     resd 1  ; callback service address or s.r.b. address
                             <1>                    ; (s.r.b. addr is physical, cbs addr is virtual)
                             <1>
00019CC8 <res 00000001>      <1> audio_bps:   resb 1  ; selected mode: 8 bit, 16 bit
00019CC9 <res 00000001>      <1> audio_stmo:  resb 1 ; selected mode: mono /stereo
00019CCA <res 00000002>      <1> audio_freq:  resw 1 ; sampling rate
                             <1>
                             <1> ; 21/04/2017
00019CCC <res 00000001>      <1> audio_play_cmd: resb 1  ; Play/Stop command (1 = play, 0 = stop)
                             <1> audio_civ: ; 28/05/2017 ; Current Buffer Index (AC'97)
00019CCD <res 00000001>      <1> audio_flag_eol:    resb 1  ; End of Link status (vt8233, EOL/FLAG)
                             <1>
                             <1> audio_master_volume:
00019CCE <res 00000001>      <1> audio_master_volume_l: resb 1 ; sound volume (lineout) left channel
00019CCF <res 00000001>      <1> audio_master_volume_r: resb 1 ; sound volume (lineout) right channel
                             <1>
                             <1> alignb 4
                             <1> ; 28/05/2017
                             <1> ; AC'97 Audio Controller Base Adress Registers
00019CD0 <res 00000002>      <1> NAMBAR:            resw 1 ; Native Audio Mixer Base Address
00019CD2 <res 00000002>      <1> NABMBAR:     resw 1 ; Native Audio Bus Mastering Base Address
                             <1>
                             <1> ;alignb 4
                             <1> ; 21/04/2017
00019CD4 <res 00000400>      <1> audio_bdl_buff:    resd 32*8  ; VT8233 (AC97) BDL Buffer Size
                             <1> ; 12/05/2017
0001A0D4 <res 00000004>      <1> base_addr:   resd 1 ; 'direct_memory_access' (memory.s)
                             <1>
                             <1> ; 28/08/2017
                             <1> ; 20/08/2017
0001A0D8 <res 00000001>      <1>             resb 1  ;
0001A0D9 <res 00000001>      <1> dma_user:    resb 1 ; user number for sysdma
0001A0DA <res 00000001>      <1> dma_channel: resb 1 ; dma channel for sysdma
0001A0DB <res 00000001>      <1> dma_mode:    resb 1  ; dma mode for sysdma
0001A0DC <res 00000004>      <1> dma_addr:    resd 1 ; dma buffer physical addr for sysdma
0001A0E0 <res 00000004>      <1> dma_size:    resd 1  ; dma buffer size (in bytes) for sysdma
0001A0E4 <res 00000004>      <1> dma_start:   resd 1  ; dma start address for sysdma
0001A0E8 <res 00000004>      <1> dma_count:   resd 1  ; dma count (in bytes) for sysdma
                             <1>
0001A0EC <res 00005F14>      <1> alignb 65536
                             <1> ; 09/08/2017
                             <1> ; 12/05/2017
00020000 <res 00010000>      <1> sb16_dma_buffer: resb 65536 ; DMA buffer for sb16 audio playing.
                                 ; 24/01/2016
                                 %include 'ubss.s'  ; UNINITIALIZED KERNEL (USER) DATA
                             <1> ; ****************************************************************************
                             <1> ; TRDOS386.ASM (TRDOS 386 Kernel - v2.0.0) - UNINITIALIZED USER DATA : ubss.s
                             <1> ; ----------------------------------------------------------------------------
                             <1> ; Last Update: 28/02/2017
                             <1> ; ----------------------------------------------------------------------------
                             <1> ; Beginning: 24/01/2016
                             <1> ; ----------------------------------------------------------------------------
                             <1> ; Assembler: NASM version 2.11 (trdos386.s)
                             <1> ; ----------------------------------------------------------------------------
                             <1> ; Derived from 'Retro UNIX 386 Kernel - v0.2.1.0' source code by Erdogan Tan
                             <1> ; ux.s (04/12/2015)
                             <1> ; ****************************************************************************
                             <1>
                             <1> ; Retro UNIX 386 v1 Kernel - ux.s
                             <1> ; Last Modification: 04/12/2015
                             <1> ;
                             <1> ; ///////// RETRO UNIX 386 V1 SYSTEM DEFINITIONS ///////////////
                             <1> ; (Modified from
                             <1> ;     Retro UNIX 8086 v1 system definitions in 'UNIX.ASM', 01/09/2014)
                             <1> ; ((UNIX.ASM (RETRO UNIX 8086 V1 Kernel), 11/03/2013 - 01/09/2014))
                             <1> ; ----------------------------------------------------------------------------
                             <1> ; Derived from UNIX Operating System (v1.0 for PDP-11)
                             <1> ; (Original) Source Code by Ken Thompson (1971-1972)
                             <1> ; <Bell Laboratories (17/3/1972)>
                             <1> ; <Preliminary Release of UNIX Implementation Document>
                             <1> ; (Section E10 (17/3/1972) - ux.s)
                             <1> ; ****************************************************************************
                             <1>
                             <1> alignb 2
                             <1>
                             <1> inode:
                             <1>       ; 11/03/2013.
                             <1>       ;Derived from UNIX v1 source code 'inode' structure (ux).
                             <1>       ;i.
                             <1>
00030000 <res 00000002>      <1>       i.flgs:       resw 1
00030002 <res 00000001>      <1>       i.nlks:       resb 1
00030003 <res 00000001>      <1>       i.uid:  resb 1
                             <1>         ;i.size:  resw 1 ; size
00030004 <res 00000002>      <1>       resw 1 ; 29/04/2016
00030006 <res 00000010>      <1>       i.dskp:       resw 8 ; 16 bytes
00030016 <res 00000004>      <1>       i.ctim:       resd 1
0003001A <res 00000004>      <1>       i.mtim:       resd 1
0003001E <res 00000002>      <1>       i.rsvd:  resw 1 ; Reserved (ZERO/Undefined word for UNIX v1.)
                             <1>
                             <1> I_SIZE      equ $ - inode
                             <1>
                             <1> process:
                             <1>       ; 19/12/2016
                             <1>       ; 21/05/2016
                             <1>       ; 19/05/2016 - TRDOS 386 (TRDOS v2.0)
                             <1>       ; 06/05/2015 - Retro UNIX 386 v1
                             <1>       ; 11/03/2013 - 05/02/2014 (Retro UNIX 8086 v1)
                             <1>       ;Derived from UNIX v1 source code 'proc' structure (ux).
                             <1>       ;p.
```

```
                            <1>
00030020 <res 00000020>     <1>         p.pid:   resw nproc
00030040 <res 00000020>     <1>         p.ppid:  resw nproc
00030060 <res 00000020>     <1>         p.break: resw nproc
00030080 <res 00000010>     <1>         p.ttyc:  resb nproc ; console tty in Retro UNIX 8086 v1.
00030090 <res 00000010>     <1>       p.waitc: resb nproc ; waiting channel in Retro UNIX 8086 v1.
000300A0 <res 00000010>     <1>       p.link:       resb nproc
000300B0 <res 00000010>     <1>       p.stat:       resb nproc
                            <1>
                            <1>       ; 06/05/2015 (Retro UNIX 386 v1 feature only !)
000300C0 <res 00000040>     <1>       p.upage: resd nproc ; Physical address of the process's
                            <1>                           ; 'user' structure
                            <1>       ; 21/05/2016
                            <1>       ; 19/05/2016 (TRDOS 386 feature only!)
00030100 <res 00000010>     <1>       p.timer: resb nproc ; number of timer events of the processs
                            <1>
                            <1>       ; 19/12/2016
00030110 <res 00000040>     <1>       p.tcb: resd nproc ; timer callback service address (if > 0)
                            <1>
                            <1> P_SIZE      equ $ - process
                            <1>
                            <1> ; fsp table (original UNIX v1)
                            <1> ;
                            <1> ;Entry
                            <1> ;          15                                      0
                            <1> ;  1     |---|--------------------------------------|
                            <1> ;        |r/w|       i-number of open file          |
                            <1> ;        |---|--------------------------------------|
                            <1> ;        |               device number              |
                            <1> ;        |------------------------------------------|
                            <1> ;    (*) | offset pointer, i.e., r/w pointer to file |
                            <1> ;        |------------------------------------------|
                            <1> ;        |  flag that says    | number of processes  |
                            <1> ;        |   file deleted     | that have file open  |
                            <1> ;        |------------------------------------------|
                            <1> ;  2     |                                          |
                            <1> ;        |------------------------------------------|
                            <1> ;        |                                          |
                            <1> ;        |------------------------------------------|
                            <1> ;        |                                          |
                            <1> ;        |------------------------------------------|
                            <1> ;  3     |                                          |
                            <1> ;        |                                          |
                            <1> ;
                            <1> ; (*) Retro UNIX 386 v1 modification: 32 bit offset pointer
                            <1>
                            <1>
                            <1> ; 15/04/2015
00030150 <res 000001F4>     <1> fsp:   resb nfiles * 10 ; 11/05/2015 (8 -> 10)
00030344 <res 00000002>     <1> idev:  resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
00030346 <res 00000002>     <1> cdev:    resw 1 ; device number is 1 byte in Retro UNIX 8086 v1 !
                            <1> ; 18/05/2015
                            <1> ; 26/04/2013 device/drive parameters (Retro UNIX 8086 v1 feature only!)
                            <1> ; 'UNIX' device numbers (as in 'cdev' and 'u.cdrv')
                            <1> ;     0 -> root device (which has Retro UNIX 8086 v1 file system)
                            <1> ;     1 -> mounted device (which has Retro UNIX 8086 v1 file system)
                            <1> ; 'Retro UNIX 8086 v1' device numbers: (for disk I/O procedures)
                            <1> ;     0 -> fd0 (physical drive, floppy disk 1), physical drive number = 0
                            <1> ;     1 -> fd1 (physical drive, floppy disk 2), physical drive number = 1
                            <1> ;     2 -> hd0 (physical drive, hard disk 1), physical drive number = 80h
                            <1> ;     3 -> hd1 (physical drive, hard disk 2), physical drive number = 81h
                            <1> ;     4 -> hd2 (physical drive, hard disk 3), physical drive number = 82h
                            <1> ;     5 -> hd3 (physical drive, hard disk 4), physical drive number = 83h
00030348 <res 00000001>     <1> rdev:  resb 1 ; root device number ; Retro UNIX 8086 v1 feature only!
                            <1>               ; as above, for physical drives numbers in following table
00030349 <res 00000001>     <1> mdev:  resb 1 ; mounted device number ; Retro UNIX 8086 v1 feature only!
                            <1> ; 15/04/2015
0003034A <res 00000001>     <1> active:      resb 1
0003034B <res 00000001>     <1>        resb 1 ; 09/06/2015
0003034C <res 00000002>     <1> mnti:  resw 1
0003034E <res 00000002>     <1> mpid:  resw 1
00030350 <res 00000002>     <1> rootdir: resw 1
                            <1>
                            <1> ; 21/05/2016 - TRDOS 386 (TRDOS v2.0) - priority levels, 3 run queues
                            <1> runq:
00030352 <res 00000002>     <1> runq_event:   resw 1 ; high priority, 'run for event'            ; 2
00030354 <res 00000002>     <1> runq_normal:  resw 1 ; normal/regular priority, 'run as reqular' ; 1
00030356 <res 00000002>     <1> runq_background: resw 1 ; low priority, 'run on background'         ; 0
                            <1> ;
00030358 <res 00000001>     <1> imod:  resb 1
00030359 <res 00000001>     <1> smod:  resb 1
0003035A <res 00000001>     <1> mmod:  resb 1
0003035B <res 00000001>     <1> sysflg:      resb 1
                            <1>
                            <1> alignb 4
                            <1>
                            <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 <res 00000004>                 <1>      u.sp:    resd 1 ; esp (kernel stack at the beginning of 'sysent')
00030360 <res 00000004>                 <1>      u.usp:   resd 1 ; esp (kernel stack points to user's registers)
00030364 <res 00000004>                 <1>      u.r0:    resd 1 ; eax
00030368 <res 00000002>                 <1>      u.cdir:         resw 1
0003036A <res 0000000A>                 <1>      u.fp:    resb 10
00030374 <res 00000004>                 <1>      u.fofp:         resd 1
00030378 <res 00000004>                 <1>      u.dirp:         resd 1
0003037C <res 00000004>                 <1>      u.namep:  resd 1
00030380 <res 00000004>                 <1>      u.off:   resd 1
00030384 <res 00000004>                 <1>      u.base:         resd 1
00030388 <res 00000004>                 <1>      u.count:  resd 1
0003038C <res 00000004>                 <1>      u.nread:  resd 1
00030390 <res 00000004>                 <1>      u.break:  resd 1 ; break
00030394 <res 00000002>                 <1>      u.ttyp:         resw 1
                                        <1>      ; 10/01/2017 (TRDOS 386, relocation and dword alignment)
                                        <1>      ; tty number (rtty, rcvt, wtty)
00030396 <res 00000001>                 <1>      u.ttyn:         resb 1 ; 28/07/2013 - Retro Unix 8086 v1 feature only !
00030397 <res 00000001>                 <1>      u.resb:   resb 1 ; 10/01/2017 (TRDOS 386, temporary)
00030398 <res 00000010>                 <1>      u.dirbuf: resb 16 ; 04/12/2015 (10 -> 16)
                                        <1>      ;u.pri:         resw 1 ; 14/02/2014
000303A8 <res 00000001>                 <1>      u.quant:  resb 1 ; Retro UNIX 8086 v1 Feature only ! (uquant)
000303A9 <res 00000001>                 <1>      u.pri:   resb 1 ; Modification: 21/05/2016 (priority levels: 0, 1, 2)
000303AA <res 00000002>                 <1>      u.intr:         resw 1
000303AC <res 00000002>                 <1>      u.quit:         resw 1
                                        <1>      ;u.emt:         resw 1 ; 10/10/2013
                                        <1>      ;u.ilgins: resw 1 ; 10/01/2017
000303AE <res 00000002>                 <1>      u.cdrv:         resw 1 ; cdev
000303B0 <res 00000001>                 <1>      u.uid:   resb 1 ; uid
000303B1 <res 00000001>                 <1>      u.ruid:         resb 1
000303B2 <res 00000001>                 <1>      u.bsys:         resb 1
000303B3 <res 00000001>                 <1>      u.uno:   resb 1
000303B4 <res 00000004>                 <1>        u.upage:  resd 1 ; 16/04/2015 - Retro Unix 386 v1 feature only !
000303B8 <res 00000004>                 <1>      u.pgdir:  resd 1 ; 09/03/2015 (page dir addr of process)
000303BC <res 00000004>                 <1>      u.ppgdir: resd 1 ; 06/05/2015 (page dir addr of the parent process)
000303C0 <res 00000004>                 <1>      u.pbase:  resd 1 ; 20/05/2015 (physical base/transfer address)
000303C4 <res 00000002>                 <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 <res 00000001>                 <1>      u.kcall:  resb 1 ; The caller is 'namei' (dskr) or 'mkdir' (dskw) sign
000303C7 <res 00000001>                 <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 <res 00000004>                 <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 <res 00000004>                 <1>      u.pfcount: resd 1 ; page fault count for (this) process (for sys geterr)
                                        <1>            ; 19/12/2016 (TRDOS 386)
000303D0 <res 00000004>                 <1>      u.tcb:   resd 1 ; Timer callback address/flag which will be used by timer int
                                        <1>            ; 13/01/2017 (TRDOS 386)
000303D4 <res 00000001>                 <1>      u.t_lock: resb 1 ; Timer interrupt (callback) lock (unlocked by 'sysrele')
000303D5 <res 00000001>                 <1>      u.t_mode: resb 1 ; running mode during timer interrupt (0= system, 0FFh= user)
                                        <1>            ; 26/02/2017 (TRDOS 386)
000303D6 <res 00000001>                 <1>      u.irqc:        resb 1  ; Count of IRQ callback services (IRQs in use)
                                        <1>            ; 28/02/2017 (TRDOS 386)
000303D7 <res 00000001>                 <1>      u.irqwait: resb 1 ; IRQ waiting for callback service flag (IRQ number, If > 0)
000303D8 <res 00000001>                 <1>      u.r_lock:  resb 1 ; 'IRQ callback service is in progress' flag (IRQ lock)
000303D9 <res 00000001>                 <1>      u.r_mode:  resb 1 ; running mode during hadware interrupt
                                        <1>            ; 27/02/2017 (TRDOS 386)
000303DA <res 00000001>                 <1>      u.fpsave: resb 1  ; TRDOS 386, 'save/restore FPU registers' flag
000303DB <res 00000001>                 <1> alignb 4
000303DC <res 0000005E>                 <1>      u.fpregs: resb 94 ; 94 byte area for saving and restoring FPU registers
                                        <1>
0003043A <res 00000002>                 <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 <res 00000004>                 <1> pcore:  resd 1  ; physical start address of user's memory space (for sys exec)
00030440 <res 00000004>                 <1> ecore:  resd 1  ; physical address of user's stack/last page (for sys exec)
00030444 <res 00000004>                 <1> nbase:      resd 1 ; physical base address for 'namei' & 'sysexec'
00030448 <res 00000002>                 <1> ncount: resw 1     ; remain byte count in page for 'namei' & 'sysexec'
0003044A <res 00000002>                 <1> argc: resw 1 ; argument count for 'sysexec'
0003044C <res 00000004>                 <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 <res 00000001>                 <1> rw:    resb 1 ;; Read/Write sign (iget)
                                        <1>
                                        <1> ;alignb 4
                                        <1>
                                        <1> ; 24/04/2016
00030451 <res 00000004>                 <1> ii:         resd 1 ; first cluster of the program file
00030455 <res 00000004>                 <1> i.size:            resd 1 ; size of the program file
                                    3649
00030459 <res 00000003>                     alignb 4
                                    3651
                                    3652     ; 23/05/2016 (TRDOS 386)
                                    3653     ; 14/10/2015 (Retro UNIX 386 v1, 'unix386.s')
0003045C <res 00000004>                     cr3reg:      resd 1  ; cr3 register content at the beginning of the timer
                                    3655                          ; (or RTC) interrupt handler.
                                    3656
                                    3657     ; 10/12/2016 (callback)
                                    3658     ; 10/06/2016
                                    3659     ; 19/05/2016
```

```
                                ; 18/05/2016 - TRDOS 386 feature only !
00030460 <res 00000100>         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 <res 00000002>         swpq_count: resw 1 ; count of pages on the swap queue
00030562 <res 00000004>         swp_drv:    resd 1 ; logical drive description table address of the swap drive/disk
00030566 <res 00000004>         swpd_size:  resd 1 ; size of swap drive/disk (volume) in sectors (512 bytes).

0003056A <res 00000004>         swpd_free:  resd 1 ; free page blocks (4096 bytes) on swap disk/drive (logical)
0003056E <res 00000004>         swpd_next:  resd 1 ; next free page block
00030572 <res 00000004>         swpd_last:  resd 1 ; last swap page block

00030576 <res 00000002>         alignb 4

                                ; 10/07/2015
                                ; 28/08/2014
00030578 <res 00000004>         error_code:  resd 1
                                ; 29/08/2014
0003057C <res 00000004>         FaultOffset:       resd 1
                                ; 21/09/2015
00030580 <res 00000004>         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 <res 00000008>               resb   8
                                readi_buffer:
0003058C <res 00000200>               resb   512
0003078C <res 00000008>               resb   8
                                writei_buffer:
00030794 <res 00000200>               resb   512
                                ; 24/10/2016
00030994 <res 00000008>               resb   8
                                rw_buffer:
0003099C <res 00000800>               resb   2048  ; general purposed, r/w sector buffer

                                %if 1
                                ; 17/01/2021
0003119C <res 00000080>         edid_info:  resb 128 ; VESA EDID (monitor capabilities) info
                                ; 28/11/2020
0003121C <res 00000001>         pmi32:              resb 1 ; (>0) use VESA VBE3 protected mode calls
0003121D <res 00000001>         vbe_mode_x:  resb 1  ; VESA VBE3 video bios mode set options
0003121E <res 00000002>         video_mode:  resw 1 ; VESA VBE3 video mode (with option flags)
                                ; 30/11/2020
00031220 <res 00000004>         vbe3bios_addr:     resd 1 ; new (writable mem) address of VBE3 bios
                                ; 02/12/2020
00031224 <res 00000004>         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 <res 00000002>         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 00000010>                     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 00000044>                     resb 68      ; mode + 66 byte VESA vbe3 mode info (4F01h)
                                %endif

                                ; 05/01/2021
0003127E <res 00000001>         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)
                                                    ; bit 0 - select 8x8 user font (sysvideo)
0003127F <res 00000001>                     resb 1 ; 19/01/2021
                                ; 17/01/2021
00031280 <res 00000001>         srvsf:              resb 1 ; 'save restore video state' permission flag
                                ; 18/01/2021
00031281 <res 00000001>         srvso:              resb 1 ; video state buffer save/restore option
00031282 <res 00000004>         VideoStateID:       resd 1 ; used to verify state saved by same prog
                                ; 29/01/2021
00031286 <res 00000002>         v_width:    resw 1 ; screen (display page) width
00031288 <res 00000001>         v_ops:              resb 1 ; 'sysvideo' graphics data transfer option
00031289 <res 00000001>         v_bpp:              resb 1 ; bits per pixels ('sysvideo')
0003128A <res 00000004>         v_mem:              resd 1 ; video memory ('sysvideo')
0003128E <res 00000004>         v_siz:              resd 1 ; video page size ('sysvideo')
00031292 <res 00000004>         v_str:              resd 1 ; window start adress ('sysvideo')
00031296 <res 00000004>         v_end:              resd 1 ; window end (end+1) adress ('sysvideo')
                                ; 31/01/2021
```

```
3764                                    ; 01/01/2021
3765                                    ;maskbuff:     ;resd 1    ; user's bitmask buffer addr ('sysvideo')
3766 0003129A <res 00000004>           maskcolor:   resd 1 ; VGA/SVGA pixel mask color ('sysvideo')
3767                                    ; 27/02/2021
3768 0003129E <res 00000004>           pixcount:    resd 1 ; pixel count ('sysvideo' window ops)
3769                                    ; 02/02/2021
3770 000312A2 <res 00000008>           buffer8:     resd 2 ; 8 bytes small buffer for 'sysvideo'
3771
3772                                    bss_end:
3773
3774                                    ; 27/12/2013
3775                                    _end:  ; end of kernel code
```